SECURITY NOW!

Transcript of Episode #454

Certificate Revocation Part 2

Description: After catching up with the week's security events, Leo and I continue and complete our examination of the history and present operation of security certificate revocation. With last week's theory behind us, this week we examine the current practice and implementation of certificate revocation.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-454.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-454-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve has all the security updates. Microsoft blinks. XP updates? Perhaps. We'll also talk about the huge kerfuffle on the Internet over certificate revocation. Part 2 of Certificate Revocation, and Steve addresses some of the criticism he's received for his position. It's coming up next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 454, recorded May 6th, 2014: Certificate Revocation Part 2.

It's time for Security Now!, the show that protects you and your loved ones online; your privacy, too. And this is the guy who does it, my Explainer in Chief, Mr. Steven Gibson. Today we should call you Steven "Tiberius" Gibson. He's our commander on the bridge on the Starship Security Now!. So we're going to - hi, Steve.

Steve Gibson: Hi. I'm trying to find my, and it seems to have - oh, there it is, my stopwatch, so that I can start it and kind of keep an eye on what's going on. Hey. Yes, today we're going to wrap up our conversation about certificate revocation. Of course all of our listeners know, anyone who has joined us in the last week, that is, that last week we laid an absolutely thorough theoretical foundation, that is, the idea being that certificates create a system that asserts a way for us to trust what another server is saying by using a common entity that we both trust. And so we're trusting that common entity's assertion about someone we don't know.

The problem is that those assertions last two or three years. And so what happens if we need to revoke that assertion? Of course that was the foundation we laid. Today I want to talk about what I've learned during these last three weeks. I am done now and back working on SQRL, to everyone who's interested in SQRL's delight. But I learned a huge amount about the state of the industry, the state of the various browsers and operating platforms, mobile and desktop, that we use. There was actually a - I guess a scuffle wouldn't put it too strongly.

Leo: A kerfuffle?

Steve: A kerfuffle between me and Google, believe it or not, because one of the things that arose early in this was that Chrome blocked the revoked.grc.com domain that I set up specifically as a test. And they did it by adding a special case to their revocation, just for GRC. And that actually...

Leo: Oh, wow.

Steve: Yeah, I know.

Leo: Aren't you special.

Steve: Uh-huh. Well, see, the point was that because - the point is that Chrome's revocation actually doesn't work. It's broken. And of course Adam Langley, who decides these things, has been famously telling people, turn it off, don't bother, all it does is slow things down. They're the only browser in the industry that does that. The other ones - Windows, Safari, Firefox, all the other browsers - have it enabled so that revocations can be detected. But because Chrome's actually doesn't work, they had to make a manual override, essentially, because people were wondering why revoked.grc.com wasn't revoked. And in fact also the other one that was on the radar was CloudFlareChallenge.com. Remember that the CloudFlare guys put up a challenge to see whether hackers could actually obtain the keys to their server. The hackers did on two different cases. And then they revoked the certificate. Well, again, that would have shown that Chrome wasn't honoring revocation.

So those two domains were added manually in Chrome's own private list. They call it a CRLSet. I have a page that's up. And Adam's not happy with me; but, you know, sorry. They kept telling people that, oh, yeah, if a certificate gets revoked, our system picks it up, and users are protected. Which is completely nonsense. And so I had to show the emperor that we really knew that he wasn't wearing any clothes. We were going to stop flattering his wardrobe because then he would be disinclined to purchase one.

Leo: Did he respond to you directly at any point?

Steve: Oh, yeah. He did a blog posting. I mean, Ars Technica picked this up and did a story on it. They all took the Heartbleed angle because one of the things I noted was that none of the 140,000 certificates that were deliberately revoked due to Heartbleed are acknowledged by Chrome. So no protection from the known potentially stolen certificates in Chrome. So, yeah, there it is.

Leo: Yeah.

Steve: Yeah. And so I put together over the course of several days a comprehensive analysis where I looked at the number of revocations to Chrome. Chrome, the problem is that, well, what I think actually - well, we'll get into this in the podcast because that's what I want to talk about is the mobile platforms, the desktop platforms, and what the different browsers do. So today, where last week was theory, today is practice. And I'm going to cover a little bit more than I did last week why this is hard.

It turns out that we've got years of evolution. We've got all kinds of backward compatibility problems and certificate fields that are overloaded, meaning that they're being asked to do multiple duty. And so building the software to actually check this is difficult. And so it's something that we're still working towards as an industry. And then we're actually going to talk about whether it matters because I think that's worth asking, too.

Leo: Ah. Actually always the most important question, isn't it, really.

Steve: Yeah.

Leo: All right. Well, we'll - boy, this is going to be a good one. Looks like a little controversy going on.

Steve: Oh, you know me, Leo.

Leo: And I missed it all. I don't know why. I guess I wasn't reading the right pages or something. But we'll talk about it in a second, if you missed it, too.

Steve: Yeah. So we're going to talk about Microsoft patching XP after all. A well-known...

Leo: Yeah. They blinked.

Steve: A well-known vulnerability in OpenID and OAuth, which has been breathlessly rediscovered, I think mostly because it was given a cool name, which seems to be important these days. You know, Heartbleed, how much better does it get than that? But this is not Heartbleed. This is Covert Redirect. It's like, oh.

Leo: Ooh.

Steve: Then a hacker discovered that older iPhones aren't encrypting email attachments. I don't know if Rene talked about this in the previous podcast.

Leo: He did, yeah, he did. And I'm...

Steve: Yeah, he did also a nice explanation over on iMore. The U.S. government has begun pilot testing their somewhat worrisome Universal CyberID. Of course the EFF is not happy about this, and we'll talk about that. And then, as I said, we'll go into the practice side of the theory and practice of certificate revocation.

So first up is, to everyone's, well, I guess it was everyone's surprise because we had been scared so much by Microsoft telling us that we had to upgrade. Microsoft pushed, after the podcast last week, an out-of-cycle update for Internet Explorer to fix this zero-day flaw that we talked about. Remember, I blogged about it. I blogged some registry changes that could be pulled off. And what I wrote, kind of tongue-in-cheek, was that Microsoft rushed an out-of-cycle update for Internet Explorer and, in their haste, apparently forgot to exclude updates for IE versions 6 through 8 on Windows XP.

Leo: Oh, you mean they did it by accident?

Steve: It's like - oh, no, no, I'm just teasing. It's like, oh...

Leo: I'm sure they thought long and hard about this.

Steve: We forgot to remove that. We told everybody no more of those. Even though we have them, we're not going to give them to them because, you know, we charge for that now. Anyway, what they did say was: "We have made the decision to issue a security update for Windows XP users. Windows XP is no longer supported by Microsoft, and we continue to encourage customers to migrate to a modern" - instead of that old antique - "to a modern operating system such as Windows 7 or 8.1. Additionally, customers are encouraged to upgrade to the latest version of Internet Explorer, IE11." So anyway, yes. Everybody got updated, even IE6, that crusty old thing. So we'll see what happens.

As you said, Leo, they blinked. And they did the right thing. In a case like this, the danger was real. It was being exploited. And it's also telling that it was necessary to reboot XP. That is, and certainly you remember the - I don't even think we were doing the podcast back in the antitrust days, when Microsoft would say, oh, no, no, we can't remove IE from Windows. It's part of the operating system. And people were like, what? That's the screwiest thing we've ever heard anyone say. You know, you have your OS, and you've got your browser. Oh, no, no, we glued them all together, and we can't take it out.

This is when, you know, the original Mozilla was arguing that, oh, wait a minute, it was anticompetitive for Microsoft to be putting in a browser and creating a disincentive for anyone to use alternate browsers. Apparently giving them away free and making them good and speedy, as Google has done with Chrome, and Firefox has done, ultimately won, especially when everyone understood that IE was such a security disaster for so long during its first half of its life.

But anyway, so it is in fact the case that this involved fixing parts of what you could say was the OS, only because Microsoft has maintained this weird entanglement of IE and Explorer such that you just can't, like, shut down Explorer and fix it and then start it up again with a new version, the way the cross-platform browsers like Chrome and Firefox allow you to.

Okay. So I got lots of tweets from people because this made the news. I think CNET may have been the first to report it. And among the security community we were just sort of rolling our eyes. It's like, oh, what? And this was a presentation made by a student somewhere about his discovery of what he named "Covert Redirect." And again, how can you not love that name? But it created, I mean, an amazing amount of press over the idea...

Leo: The kid's smart. That's how you get attention. Just a good name.

Steve: Yeah, exactly. That's what you need. OAuth and OpenID, which we've covered in depth in the podcast, and I'm not going to go back into it in great detail, mostly because this doesn't warrant it. But anyone who's interested, you can search - go to GRC.com/sn, and there's a search box. Search for "OAuth," and you'll find a podcast where we describe it in detail [SN-266]. I've essentially been, every time we talk about it, and you'll remember this, Leo, I say, oh, just wait. This thing, this protocol, this authentication method is just begging to be exploited. And this predates SQRL, so this is not me being jealous of OAuth and OpenID.

The problem is this is the new hip way of logging in, where you go to a site that doesn't know you, where you haven't been, and it says create an account, which of course no one wants to do. We call that "high friction." Or log in with your Facebook ID. Log in with your Google ID. Log in with your Twitter, you know, whatever. Log in with an existing relationship you have somewhere else.

The way that's done is you say, oh, good, I don't have to create an account here. You click one of those buttons. That button redirects your browser to the third party that's going to authenticate you. And essentially that's where you see things like this application, and it normally says it, wants the following things, wants to know these things about you or do these things. If that's okay, then - and oftentimes you're already statically logged in over at Google or Facebook or something, so it may just be a matter of approving. Or, if you're not currently logged in, you log in using those credentials of Google or Facebook. Then you say, yes, I approve, and you go bouncing back.

Well, this whole idea of bouncing your browser around between sites is frightening. And all it takes is for you to go to a site which is sketchy or spoofing a legitimate site that looks like somewhere you would want to log in, and you say, yes, log me in using these credentials. Well, that site bounces you somewhere else, and you haven't noticed, like that the name is spelled wrong or that it's PayPal.ru or something, rather than .com, which you're expecting. So it's easy to mistake that. But it's also the case that you don't see the URL that you're clicking on. You're just clicking on friendly-looking shadowed amazing glowing button. And behind the scenes is this plumbing.

Now, this whole Covert Redirect business was already described in the, I mean the vulnerability, in the IETF's OAuth Specification Section 4.2.4 on page 22. So none of this is new. This is just, you know, someone rediscovered it. And rediscovery happens. I'm sure he thought this was original. The problem is that not all authenticators like Google or Facebook or Twitter are set up with limited URLs that they will redirect their results to. That is, there are these things called "open redirectors," where they'll just authenticate anybody who comes along. And that's exactly the scenario that I just described, where you've got some spoofed site at a non-authenticate URL. Or you could have a valid site, but something managed to poison the page. Because, again, you're not seeing the URL you're clicking on, so you don't know where you're going. You're just pressing a nicely colored button, and it's all happening behind the scenes.

So what can be done is that valid sites specify exactly the URL that they want the redirection, the post-authentication bounce-back to go to. It's not necessary to do that when you establish a relationship with a Google or a PayPal or a Facebook for OpenAuth authentication. But arguably it should be required. Right now it's a little too loose. Again, you know, to make it easy, and therein lies vulnerability.

So when a site wants to host OAuth authentication, they should - it's like a firewall. They should explicitly specify that redirection from their application can only go back to this URL. And if that were done, then this whole problem goes away. And it's all spelled out in the spec. So nothing new to see here. This doesn't, I mean, it's good to remind people of this because this is still a problem.

Again, it's very difficult to bounce users around between domains and not have bad things happen. I still expect that we're going to end up seeing a major exploit of this at one point, you know, after it becomes really popular, so that it could do more damage. But it is absolutely possible for the origin site to prevent their site from being abused by specifying to the people that they're using to authenticate their visitors exactly what the URL is, or technically the URI, that is used to come back to them. And if that's done, then you can't be taken anywhere else. You can't be bounced through yet another server that's able to obtain your credentials. See, that's what can happen. They can end up with access to your Google account and access to the account that you were authenticating to. So it's a form of man-in-the-middle attack on OAuth. And it's not good.

What Rene talked about, I'm sure on MacBreak Weekly was something that caught a lot of people's attention after we had done our triple-header podcast on how amazingly secure iOS and the iPhones are. And it was a hacker, Andreas Kurtz, who wrote on April 23rd, he said: "What Apple Missed to Fix in iOS 7.1.1," his point being that they knew about it in 7.1. He had informed them. And they sounded like they sort of didn't take him very seriously. It's like, eh, yeah, well, okay. But when they didn't fix it in 0.1.1, he decided, okay, I'm going to make a little more noise. And that got their attention.

So what Andreas wrote was: "A few weeks ago, I noticed that email attachments within the iOS 7 MobileMail app were not protected by Apple's data protection mechanisms. Clearly, this is contrary to Apple's claims that data protection 'provides an additional layer of protection for email messages attachments.'"

Andreas wrote: "I verified this issue by restoring an iPhone 4" - and that's the first key. And he says in parens "(GSM) device" - meaning he did not do it on a more recent phone, where it doesn't work - "to the most recent iOS versions, 7.1 and then 7.1.1, and setting up an IMAP email account, which provided me with some test emails and attachments. Afterwards, I shut down the device" - meaning the phone - "and accessed the file system using well-known techniques - DFU mode, custom ramdisk, SSH over usbmux and so forth." He says, "Finally, I mounted the iOS data partition and navigated to the actual email folder. Within this folder, I found all attachments accessible without any encryption or restriction..."

Now, this appears to be true. And I do think it's something that Apple missed. We'll talk about why in a second. Rene Ritchie asked Apple about it and was told, "Now we're aware of the issue and are working on a fix which we'll deliver in a future software update." So maybe 0.1.2, who knows. So Rene explained this carefully on his column at iMore.com. And so first of all it's iPhones after iPhone 4, meaning the 4s, the 5, the 5s, 5c, et cetera, that is, those having at least the Apple A5 chip are not vulnerable. It's only with the previous, the A4 and earlier chips.

My guess, and you'd have to, like, look at exactly how this happened, but it sounds like probably an honest mistake where they were moving - now they have split security architectures, or multiple, actually, security architectures. They've got - they have the very latest A6 architecture with the Secure Enclave. They've got the A5 architecture. And then they've got the earlier architectures. Clearly they want to secure them all. But as they've evolved this, things like where the encryption occurs has changed. It's gone from a fast software encryption into, now, like an inline, always-present hardware encryption.

And so I can easily see where some conditional compile sort of thing, where it's like, okay, on this we follow this path, and on this version we follow this path, and if we're doing this, we follow that path. That just could have been a greater-than symbol that should have been a greater-than-equals, that kind of thing. Just a tiny little mistake due to the fragmentation now that we're seeing over the evolution of the iOS security architecture.

So my guess is it's something like that, where it probably used to encrypt on the iPhone 4 the way they intended, when that's all they were doing. But then later on someone just made a tiny mistake as they were intending to continue supporting the old devices, but had all this fancy glittery new technology for the new ones. And the wrong code got assembled, and the software encryption for the older phones didn't survive some conditional branch in the compiler somewhere would be my guess.

Leo: Yeah, just, you know.

Steve: Yeah. But again, I'm glad he found it. I mean, and as Rene points out, he quotes our podcasts on this topic in there where I refer to it as an amazing little encrypto brick - I mean, like it's just an incredible little encryption engine - and says, but, you know, we need people keeping an eye on this to keep it that way because that's what we want. Now, an unpronounceable acronym, maybe it's NSTIC...

Leo: I like it.

Steve: NSTIC, N-S-T-I-C. I've referred to it many times because it's been on my radar. Everyone knows I'm an authentication fanatic because I really think that proving our identity on the 'Net is the thing. I mean, that's what we need as we increasingly depend upon so-called cyberspace. So NSTIC, N-S-T-I-C, is the National Strategy for Trusted Identities in Cyberspace. And remember, as soon as Stina came to the states, you know she moved Yubico over to Silicon Valley so that she'd be in the middle of things, I made sure she knew about this in case she had any interest in being involved in it because I thought any involvement that they could have would be good. So this is our government's effort, not private industry, not FIDO, not Google with their multifactor. This is yet again an entirely separate project. I'm on the mailing list. I get rah-rah email every couple months about nothing having happened again because, I mean, I think this is now we're in year three. And of course this makes everyone nervous.

In my own notes I wrote it down as: In the "What Could Possibly Be Wrong With This Idea," the U.S. government's own "National Internet ID" experiment begins. Thus the news. The New York Times picked up on it. And they used the analogy of an Internet driver's license, I think because a driver's license is a civic identity. I mean, it's a government-issued ID. And Techdirt, among others, picked upon it. And their headline was: "U.S. Government Begins Rollout of Its Driver's License for the Internet." And their subhead was: "From the seizing-the-wrong-moment department."

And they just wrote at the very beginning, they said: "An idea the government has been kicking around since 2011 is finally making its debut. Calling this move 'ill-timed' would be the most gracious way of putting it." And of course this is all post-NSA and Snowden. And now they're saying, oh, we're going to come out with a governmental issued CyberID. So, okay. Michigan and Pennsylvania are the two states where a pilot study next month will be started. And it's just intergovernmental agencies. So I don't even know if a private citizen can get one. But it's just so it's probably for employees in unnamed agencies in those two states to be burdened with one more thing they've got to worry about, just sort of to have it go, have it, you know, we know how HealthCare.gov came out, so maybe they're going to work on rolling this one out a little more gently this time, or keep it to themselves for a while.

I guess I'm of two minds about this. There are things that arguably could be valuable, theoretically. I mean, let me just couch this and make - I want to be very clear that people don't think I've completely lost my mind. But, you know, filing your taxes electronically, or voting, if there were a way to do that. Of course, our guess, all of these things are subject to horrific cyber tampering, thus the concern. But there are certainly instances where we want anonymity, and this is the reverse of that. This is apparently provable identity. But I would argue, as the Internet becomes the thing, that there are - there's some scope of need where we absolutely want to be able to assert, deliberately and with intention, and in a way that is secure, this is actually me.

Leo: Yeah. Well, any financial transaction. Right? I mean...

Steve: Well, see, there we would be doing it to our bank. This is to the government.

Leo: To the government. Let me think of what, you know, you can't do it for voting, although there has been a move to do that, to prove you are who you say you are. But that...

Steve: Well, now we're talking in the future. So imagine the future where...

Leo: Well, I hope they don't do it for voting. But I guess if you want to do online voting you would have to authenticate; right?

Steve: Yeah. I mean, I think probably healthcare and voting and maybe taxes, I mean, things that are civic, federal, probably. There may be some way of going to the post office and proving, you know...

Leo: Sure, give them an ID.

Steve: Yeah, give them a birth certificate and an ID or whatever and say I want my government-issued CyberID. I don't know how it works yet. It's the last thing I'm going to worry about because, if it ever begins to actually happen, we'll definitely cover the technology. I hope they haven't messed it up. And I think in this day and age it is just - it's just in this last few years this has happened. So they must know how to do this right. But [laughing].

Leo: Didn't Facebook used to have - for some people they would ask for a government ID. You send, like, a newspaper and a picture of yourself holding the newspaper or something, to prove you are who you - there are cases where I guess you could use a government ID for something like that.

Steve: And that's exactly my point, is I can see where you...

Leo: But you've got one. You've got a driver's license. That's a government ID.

Steve: Yeah, just hold it up and wave in front of the camera. Okay. Try to hold it still. Okay, how's that?

Leo: But we know, I mean, there have been attempts for voter ID in many states. And it's widely considered a bad idea because it disenfranchises poorer voters. So I don't know...

Steve: Right. And even the machines, they can't even make a machine that works.

Leo: That's right, that's right.

Steve: You know, it's like, oh, gosh, yeah.

Leo: It seems like a bad idea.

Steve: I mean, it's very worrisome. But I just thought, since it's coming up on the news, this is beginning to raise its head. I've referred to it many times in the past because, as I said, I get random email from them saying, oh, we're having a big conference. It's like, okay.

Leo: NSTIC. NSTIC. Well, you know, I mean, there's been - every once in a while somebody floats the idea of a national ID program, and it usually gets turned around.

Steve: Well, this is, I mean...

Leo: It would effectively be, wouldn't it.

Steve: This is moving forward. So, I mean, it may get squashed. The EFF just, I mean, if they had their way this thing would be under their heels being ground into nothing right now. So...

Leo: Well, if they're agin it, I'm agin it.

Steve: Yeah. They generally have the right perspective. I had promised our listeners some graphs of the show length over time, but the URL I was given last week, when I thought I had - there just wasn't time to get it into the notes, still doesn't work. So I think it's a listener of ours who has been doing this, so I'd love...

Leo: What's this Security Now! stats thing? What is this?

Steve: Oh, it didn't come up for me. There it is.

Leo: Oh, it's working. I just pulled it up.

Steve: Oh, there it is.

Leo: It's up.

Steve: Yeah.

Leo: Did you do this, Cyphase? Cyphase maybe did it, I don't know.

Steve: Yep, Cyphase did it.

Leo: All right. So he says it's up now.

Steve: Ah, cool. I tried it an hour ago, and it wasn't.

Leo: So the green line is a four-episode rolling mean. The red line is a 12-episode rolling mean. They're pretty correlated. And the length of the show...

Steve: Now, but there was one...

Leo: ...has been going up. Its biggest growth was in the first 200 episodes. Then it stabilized for a little longer. And then from 350 and on, it's been going up again. A lot.

Steve: Well, I mean, it can't go any higher. We've reached our limit, Leo. I think now we're going to have...

Leo: Now we're 110 minutes. That's almost - we're almost two hours now.

Steve: I think it's going to flatline.

Leo: That's pretty - thank you, Cyphase.

Steve: Although was there a second one, or only...

Leo: Yeah, there's feedback and not feedback.

Steve: Okay. That's the one that I thought was really interesting, too. Remember that the Q&As for a while were running a lot longer.

Leo: Yeah, because we'd ask questions, yeah.

Steve: Yeah, because we had all the news. Then we said, oh, crap, we've got 10 questions we've got to get to now. So they...

Leo: Not because the Q&A has gotten shorter, but for some reason the non-Q&As are getting longer. They're growing - let's put it this way. They're growing faster than the Q&As are growing.

Steve: So for anyone who hasn't seen the show notes, it's Cyphase.com/securitynowstats.

Leo: And there's his bitcoin QR code, if you'd like to thank him.

Steve: Ah, very nice.

Leo: Send him a thousandth of a bitcoin.

Steve: So that's the good news. The bad news is that "Almost Human" has been canceled.

Leo: Yeah, I saw it. They didn't even - one season. That's all I got.

Steve: They, yeah...

Leo: We liked that show.

Steve: I guess they got more episodes than "Firefly" did. But they of course famously canceled Joss Whedon's fabulous sci-fi space Western. And this one died. I really liked it. Yeah, I mean, it wasn't to die for. But it must have been expensive. I think what they look at is - and apparently this didn't get the numbers. So there was just a small following of people who really enjoyed it, but not a large enough following.

Leo: But I see you're looking to "Halt and Catch Fire" as...

Steve: Oh, yes.

Leo: I don't know if you saw the promo this week during "Mad Men" where they quoted Steve Wozniak.

Steve: Yes, I did see it. I did see it.

Leo: Something like Steve says, "I don't usually like this kind of thing, but I like this."

Steve: So I've got something in my eye. Excuse me, I'm blinking a little bit.

Leo: This is, okay, this is a show that's going to be on AMC starting in June.

Steve: Yes. I wanted to give all of our listeners a heads-up. I believe this is the story of the creation of the IBM PC, which was a...

Leo: It takes place in Texas, though. That's what confused me.

Steve: Yeah, well, see, they may have changed the names...

Leo: This was in Boca; right?

Steve: Yeah, famously Boca Raton, Florida. And so, but I don't know what else this could be. First of all...

Leo: Maybe Texas Instruments. It may also be completely made out of whole cloth. It may not be related to history. You think it's real?

Steve: I don't know. Anyway, whatever it is, we have no idea.

Leo: And apparently, what, Halt and Catch Fire is a machine code. Oh, my gosh, we've lost Steve. He's going down, folks. Do you want to take a break?

Steve: I don't know. I had to take my contact lens out.

Leo: I'm looking at Steve, and it looks as if he has recovered his sight. Are you okay?

Steve: I still wear hard contact lenses.

Leo: What?

Steve: Since that's what I started with.

Leo: Yes, me, too. But I don't...

Steve: And I'm comfortable with them.

Leo: You know what, Steve, please, do me a favor. Go to your optical professional, your optometrist. Say you want Daily Wear. Because what you do is you throw them out at the end of the day, every day.

Steve: Tried those.

Leo: You didn't like them?

Steve: I don't - no.

Leo: Because your vision wasn't as acute?

Steve: Yeah, they don't actually provide as good a correction as the...

Leo: Well, that's because hard lenses are better.

Steve: Yeah. These are gas-permeable contacts. I've been wearing them since I was like a junior. And it's funny, too, because I remember my best friend in high school, Scott Wilson, when my mom and sister were urging me to get them because I had the traditional big, thick, I mean, my eyes are very nearsighted. And I said, "So what do you think about the idea, Scott?" He says, "Oh, absolutely." Just no hesitation at all. That's what your best friend is for. It's like, oh, get rid of those glasses, Gibson. Anyway, so...

Leo: Oh, well. All right. Okay. Suffer.

Steve: Been wearing them ever since.

Leo: Suffer all you want. I'm not going to stop you.

Steve: Well, okay.

Leo: You're right, though. Acuity goes down. You're right. Acuity goes down, I agree.

Steve: 454 episodes, and I've never had a contact lens emergency in the middle of a podcast. So...

Leo: First one. I've had many.

Steve: Okay. So we don't know anything about the upcoming AMC series, beginning on Sunday, June 1st, called "Halt and Catch Fire."

Leo: Now, "halt" is a longtime, everybody knows, assembly language command. Just halt, halt the processor.

Steve: Yes. Essentially old, like original machines, it essentially - the halt instruction is a jump to yourself. So what it does is it just causes the system to stop executing instructions.

Leo: Right.

Steve: And I did some research. I was sure that the phrase had more meaning than I was able to track down on the 'Net now. Of course this predates the 'Net, and so do we. So maybe I'm remembering correctly. But I really thought I remembered something where, in the early days of the development of the PC, there actually was something that caused a serious problem, like you could do a series of instructions that would cause it to halt and overheat.

Leo: Really.

Steve: And in some cases it started to smoke or something. So...

Leo: It sounds more like a programmer joke than an actual instruction.

Steve: Well, and, see, that's all that has survived today. HCF it's supposed to be, a made-up instruction, Halt and Catch Fire. But it's like, okay. I really do think I'm remembering something, but I couldn't find it. So anyway, it looked a little maybe overdramatic. It may be more drama than techie. But who knows. I am enjoying "Silicon Valley," by the way.

Leo: Me, too. Holy cow.

Steve: A half hour. And the more an insider you are, I think the funnier it is because you realize, these really are the personalities being depicted of the loony tunes in the Valley. And many of the crazy things, like just over-the-top money being spent on parties to describe, like, some API function. It's like, what? Just would have no meaning to anybody else except a really small group.

Leo: Well, like the TechCrunch, like this week they get accepted to TechCrunch Disrupt, the startup tournament. And I think, I mean, it's real. We all know about it. In fact, Alexia Tsotsis, who works at TechCrunch, tweeted: "I think Mike Judge should be a judge at the next TechCrunch Disrupt."

Steve: And they have been renewed for a second season already.

Leo: And rightly so. Now, here's a sad note. To me, one of the best characters, and this week he was very good, is the Peter Gregory guy, the guy who - he's kind of an amalgam of angel capitalists, but I think he's mainly based on Peter Thiel. It's his last episode. The actor passed away.

Steve: No. He was young.

Leo: He was 48, and he had lung cancer when he auditioned for the role. And it took a turn for the worse. And I believe he only made five episodes, and I think we've seen the last of them. And you know what, he's brilliant in this show.

Steve: Oh, he did just a...

Leo: Borderline, you know...

Steve: Understated, yes, like ADDDDDDD.

Leo: Yeah, he's kind of "on the spectrum," as they say in the show.

Steve: Yeah.

Leo: And he's wonder- and he really had a turn, a star turn in this most recent episode. Yeah, kind of sad.

Steve: Wow. Well, I guess it'll be interesting to see how the writers work him out. I mean, they'll have...

Leo: He's fairly critical to the story.

Steve: Yeah, he's been in the center of it. So almost...

Leo: I'd rather see him than almost anybody else. I like Big Head. I like the Huli guy. There's a scene this week where they're attempting to use Skype-like objects, and everything fails. He's in Jackson Hole, Wyoming, and nothing's working.

Steve: Yup, he's got that 3D projector...

Leo: He's got a holographic projector.

Steve: For those of us who are doing video podcasts...

Leo: We know.

Steve: You know, it's like, oh, Gina's frozen again. Wait a minute. She'll reboot and come back. And, I'm sorry, Leo, I can't hear you. Would you say that - oh, it's like, oh.

Leo: We've so been there, yeah.

Steve: Anyway, so for insiders, if you're not watching it...

Leo: It's fun.

Steve: ...it's an HBO half-hour every Sunday evening, immediately after "Game of Thrones." And I guess it's before "Vice"? I think "Vice" follows it up.

Leo: "Veep," you mean, "Veep."

Steve: "Veep," sorry, "Veep," yeah.

Leo: Yeah, which is another great show.

Steve: Yeah, and that's...

Leo: My Sunday night is completely consumed by "Mad Men," "Game of Thrones," "Veep," and "Silicon Valley." Oh, and now John Oliver's got a show.

Steve: Oh, it's awful.

Leo: Yeah, it's a complete ripoff. You know, I love John Oliver deeply, and he was on "The Daily Show," and he's basically ripped off "The Daily Show." And I don't understand why they couldn't...

Steve: And not well, Leo. I haven't made it through. I've only made it about 10 minutes into either of them. And I thought I removed it from my Season Pass last week, but it came in again. And I thought, what? So I thought, okay, I'll give it a second shot. And it just, again, like maybe five minutes, it's like, oh, my god, this is bad.

Leo: Too bad, because I love John Oliver. He's very talented.

Steve: Oh, and I agree with you. He was great when he had good writers over with "The Daily Show."

Leo: But you have to admit that his - the first episode, where he interviews the former director of the NSA, General Alexander - did you see that one? First episode.

Steve: No, I deleted it. I didn't get that.

Leo: Oh, you want to see that. He's sitting across from the former head of the NSA, and he's giving him a really hard time. And he, oh, I can't remember what the punch line is. John Oliver's famous for these, like, in-your-face, very nice interviews.

Steve: I've never understood, in those interviews, if the other person realizes that they're being made fun of. It's like...

Leo: Alexander realizes it eventually. He's kind of smiling and laughing along. It's quite good. Quite good. Anyway...

Steve: Okay. So also, this is in errata, I've been meaning for weeks to mention that, after you and I talked about the electronic funds transfer concern and about setting up accounts so that, as I have, so that they're firewalled, and you cannot electronically transfer funds, and you were saying, wait a minute, that can't be right? Well, we were both right. Because it turns out that the rules and regulations for business are different from personal. There is personal protection. There is no business protection.

Leo: Ah, that's a relief.

Steve: Yes.

Leo: Okay. So, and this is kind of how the banking industry handles lack of security across the board: Don't worry. We'll pay for it if you lose anything.

Steve: Right. We don't want you coming in. We don't want to have any contact with you.

Leo: Right.

Steve: We want our computer to do this. So in return for the privilege of never having to see you, if anything goes wrong, we'll cover you.

Leo: Here's John Oliver and General Keith Alexander. I'm going to skip...

[BEGIN CLIP]

GENERAL ALEXANDER: It was wrong for me to leave that for somebody else to do.

JOHN OLIVER: Do you think that the NSA is suffering from a perception problem with the American people at the moment, bearing in mind that the answer to that is yes?

GENERAL ALEXANDER: Absolutely. You know, the first assumption is that you're collecting on the American people. And therein lies...

Leo: So he attempts to do the PR spin.

Steve: He's trying to do it straight.

Leo: Yeah.

JOHN OLIVER: Now, the target is not the American people. But it seems that too often you miss the target, and you hit the American person standing next to it going, whoa, whoa, him.

GENERAL ALEXANDER: But, see, we're not just out there gathering U.S. communications, listening to it...

[END CLIP]

Leo: It's actually really worth seeing. It's on YouTube, if you want to see it.

Steve: That sounds great.

Leo: And it is, I mean, it's the director, former director of the NSA.

Steve: Yeah.

Leo: Keith Alexander.

Steve: Yeah, I deleted it too soon.

Leo: That one thing, watch that.

Steve: The good thing is I can undelete. So I will undelete it and go find it.

Leo: Do you have a TiVo? What do you have?

Steve: Yeah, I dropped my Windows Media boxes and went back to TiVo. And I am so happy.

Leo: Me, too.

Steve: I went to the Roamio. And it's funny because I convinced my best friend to do it. And he was in Seattle. And I said, "You know, Mark, you can watch your shows on your iPad." He says, "What? No." And I got a text from him a few minutes later, "Oh, my god, it works."

Leo: It's got Slingbox-style capability for the house and...

Steve: Yup. And you're able to schedule...

Leo: Yeah. Love it.

Steve: Yeah. It's the right - and then the little Minis. In fact, I've got one right sitting next to me. So you're able to do extensions. Yeah. I tried the media box, gave it a shot for a year. And when I saw that they were getting ready to discontinue their hardware, I thought, no, no, no, I want their hardware. And so - and I also verified that, when it dies, you put a blank drive in, and it sees that it's blank, and it completely sets it up and formats it and preps it for you.

Leo: It blesses it. Oh, that's good.

Steve: And it also has an external e-something. I can't remember...

Leo: eSATA. eSATA.

Steve: Thank you, yes, an eSATA connector so you can expand...

Leo: You can add more drives. But the one I got, the Pro, has 450 hours of recording. I mean, it's like 3TB. It's plenty.

Steve: Well, you know that we got the same one. So I have...

Leo: No, I...

Steve: I also switched over to the Bluetooth remotes because I wanted to use the keyboard. And it's nice not having to aim it.

Leo: Oh, that's nice.

Steve: The normal remote is RF.

Leo: Right.

Steve: But this is actually Bluetooth. And the fact that, no matter what, no matter where you are, you just hit the Search button, and it instantly jumps you to that. So then you type...

Leo: I'm going to order that remote.

Steve: ...type in a few characters. Yeah, you just go to TiVo.com, and you choose whether it's for the Premiere or the Roamio Pro or the Mini because they have two different ones.

Leo: Yeah, I have the Pro.

Steve: Because the Mini doesn't have Bluetooth built in.

Leo: And we should mention it's a very expensive solution because you have to buy, not only the hardware, but you have to buy either a monthly subscription or, as I imagine you did, the lifetime subscription, which is several hundred bucks.

Steve: Yep. I know that I'm - yeah, it is pricey. But it's, you know, boy, it's nice. And you can undelete shows because...

Leo: I didn't know that. Because I'm deleting - the TiVo suggestions, and it records a lot because it can do up to six channels at the same time, so it's recording a ton of crap. And I mean crap. Finally I had to delete the Spanish language stations from my channel lineup...

Steve: Yup, done that.

Leo: ...because it kept recording Spanish shows. So I just said, well, I'm not going to ever watch those, so I'm taking Univision out of my lineup. But you do have - and we should say, since we're talking about this, you have to have...

Steve: Already in it.

Leo: Your supporter has to - your cable company has to support CableCARD. You need an M-card, a multistream card.

Steve: Yup. And then you can record six things at once.

Leo: So Verizon FIOS works. Some cable companies do check, of course, with your provider. And in some cases, my case, and I presume your case, we also get on-demand. I get XFINITY On Demand through it, which is really nice because then I can watch everything I want to watch. And you don't get stupid E! Entertainment reporters screaming at you the whole time you're trying to navigate through the On Demand menu.

Steve: Right, right. I did turn off all of those TiVo, the TiVo guessing what you want nonsense because it shows you a nice little percentage bar of how much space you've used. And when things accumulate, I'll flush them. Anyway, I'm really happy.

Leo: I'm afraid it's just going to churn the drive to hell. I mean, there's plenty of space, but I just don't want it to churn everything and wear out the drive fast. Good. Well, there we go. We got an unprompted review of the TiVo in there.

Steve: So SQRL, I'm back on SQRL, working on it full time. So I just wanted to let everybody know that would be moving forward. And I'm back to - I'll provide a weekly touch base. I expect it's going to go very quickly now since that whole multilingual UI system is worked out. I've got the entropy harvesting in my head, and I've conceptually got it. In two weeks we're going to talk about the challenge of harvesting entropy, which many people have been asking, hey, whatever happened to that?

Leo: Good subject, yeah.

Steve: Yeah.

Leo: Love to do that.

Steve: Because it's really so much more difficult to do it, not only, I mean, people who don't care at all, they just call their random function in whatever language they're using, which is notoriously awful. Probably we're past that. But for my application I need both high-quality randomness and intervention-proof randomness. I need attack-resistant entropy. So that means you just can't ask the operating system for it because that interface to the OS could be infected, and someone could be returning zeroes in order to affect your entropy harvesting. So anyway, in two weeks we're going to talk about really, really robust entropy harvesting, I think a really interesting topic. And I will be past it by then, but all tuned up on it.

Leo: Dr. Mom says you want to harvest entropy, get a cat. Do we - did you want to show this picture ever?

Steve: Well, yeah, put it up. It's actually an amazingly, I thought, it's the best instance of Photoshopping I've seen in, like, a long time.

Leo: Well, it makes us look pretty dang good. So who did this?

Steve: I don't know. I don't know. It's just someone who tweeted it to me this morning. And I thought, wow, you know, I mean, the lighting is a little - it's a little...

Leo: Oh, come on. Now you're being picky. This is perfect.

Steve: It really is a great...

Leo: Steve is Jean-Luc Picard, and I am his Number One, Commander Riker. They've superimposed our heads, obviously. But it looks like we're in the same room. It looks, I mean, this is quite well done.

Steve: We've got an out-of-focus Worf in the background.

Leo: Yeah, I like it.

Steve: Yeah, it really is good.

Leo: Steve and Leo, the Next Generation.

Steve: I think I'll stick it up on the Security Now! page, just for a while, at least, because people will get a kick out of it.

Leo: I think it's going to be my new profile page.

Steve: I don't know where he got the raw material. I don't think we've ever looked that good.

Leo: I know. I don't - these are amazing photos of us.

Steve: Yeah.

Leo: Very nicely done.

Steve: So thank you.

Leo: Thank you.

Steve: Okay. Speaking of thank you, Kyle Lyons. He's continuing a theme that I just want to cover for a minute. He said: "SpinRite Brings a MacBook Pro Back Up to Speed." And you'll notice that the last couple of weeks, just for whatever reason, people have been noticing that SpinRite's been improving the speed of their machines. So what Kyle wrote was: "Add me to the now typical MacBook success story. The system was taking 10 to 15 minutes to boot, and just as long..."

Leo: That's definitely slow.

Steve: There's something wrong there.

Leo: He shouldn't take that long.

Steve: "...and just as long to launch programs. The drive was reimaged to no avail. We hooked up the MacBook's drive to a Windows machine using an IDE/SATA to USB 2.0 adapter, mounted the drive to a VMware Player DOS virtual machine, and ran SpinRite on it. Four hours later, the drive and the MacBook are running like new. Thanks, Steve." So...

Leo: Wow.

Steve: Yeah. And here's what's going on. I mean, really you don't want to wait, if this is what's happening. In the early days of hard drives, error correction took longer than just rereading. That is, we didn't have high-speed, on-the-fly error correction. So the drive would try to read a sector. And if it came back with a bad CRC - Cyclic Redundancy Check - even though it had error correction information, it would take longer for the LSI chip on the drive to process it than it would for the sector to come around again. So the drive would try a few retries, thus the term "retry." It would just hopefully - it would just read it a few more times, hoping to get one that did not need correction. If that failed, then it would say, okay, fine. And it would crunch the whole 4,096 bits of data, plus the error correction stuff, through the error correction algorithm to produce what's called the "syndrome," which is an XOR mask, which is then placed appropriately. And that flips the bits that were wrong, making them right, thus correcting the data.

Now move forward to present day. Now we've got, of course, crazy speed in the drive electronics such that error correction is no longer a last resort. It's used, some would feel distressingly, with a distressing frequency, so that errors are just now commonplace. In fact, the densities have gotten so high, the drives are correcting errors more often than not. But this syndrome I mentioned is a span of bits. So think of it as it's a bit mask of bits that need to be flipped. But the power of the ECC is the number of bits long that this syndrome is. And in the old days it used to be 11. But that was the spec on the controllers before they got moved into the drives was that - the point was that, for reasons of math, you could correct any 11 bits, a burst, a burst error of up to 11 wrong bits. But 12, and you couldn't correct it.

Now they're much fancier because they're doing interleaved ECC, where you can do multiple bursts in a block and all kinds of craziness. And of course these big 4K sectors, where they really change the math, because the larger the area you're correcting, the more efficient the error correction is in terms of the amount of overhead needed for that block of data. So the logic has turned around so that error correction is being done all the time. It's on the fly. It slows nothing down. That means, if something is slowed down, it's because that is no longer correctable. You are depending upon trying again. And essentially you've got, if it's taking, for example, 10 to 15 minutes to boot, the system is being very patient because it has no choice, and it's trying to read many, many, many, many hundreds of megs, but a whole bunch of sectors are causing problems. They are beyond the on-the-fly correction. They're into the, oh, my goodness, let's hope we can get it if we ask again.

Now, of course, SpinRite famously has, essentially, you know, that's its bread and butter. That's its soul. It's able to diagnose the drive, work with a drive in that condition. And even when the drive can't finally recover, SpinRite has ways of, like, getting that one last bit and making it work. So my point is that, if anyone is noticing this happening, this is a reason to run SpinRite. What Kyle did not only sped up his drive, but it took all of those sectors - and apparently many, if it was taking this long, if he was actually - if it's enough so you can feel a slowdown, then you're on the edge, and there are many sectors on the edge. And so SpinRite pulled them all back. And now the drive is having no trouble booting. It boots up just like new because SpinRite removed, worked with the drive to remove all of those flaky sectors that were on the verge of no longer being flaky, they were being unrecoverable.

So anyway, that's a perfect use case for SpinRite in, as we say, in a preventative maintenance mode. If you ran it quarterly, it would have never gotten that bad. It's a good thing that Kyle ran it in time.

Leo: How do you run it on a Mac? Do you have to take the drive out?

Steve: Yeah. There are a couple things you can do. You can take the drive out, if you're able to. You can also boot it, if you can boot an external drive, and the Mac allows you to do that, then you can run DOS, or you can run, now, he ran VMware. But most people use virtual, what is it, VirtualBox.

Leo: VirtualBox is the free one, yeah.

Steve: VirtualBox, yes. And VirtualBox...

Leo: For this, you don't need more than that. I mean, that's...

Steve: Exactly. And then - and because you booted an external drive, you now have offline access to the internal drive. And so VirtualBox and SpinRite can reach back into your Mac and work on that drive natively. And of course, once SQRL's behind us, and we have SQRL running, then I'm running right back to SpinRite 6.1. And one of the big features of it will be native operation on the Mac. In the meantime, if you google "SpinRite Mac," you'll find people who've done how-tos and things.

Leo: There's a whole wiki entry for it.

Steve: Yeah.

Leo: Wikipedia, the SpinRite entry has a "SpinRite on Mac," give you all the details.

Steve: Neat.

Leo: Yeah. Steve Gibson, Leo Laporte, Security Now!, Revocation Part 2.

Steve: So one of the things that I want to be clear on, or I wanted to cover, that I really haven't, is why this is hard to do, why the whole certificate trust issue is difficult. It's very easy from sort of the ivory tower to say, oh, well, you have a certificate authority, and they have a self-signed certificate that they guard, and they signed an intermediate certificate, which is the one that they use, and then you have end certificates that are signed by that certificate, and now you have a chain of trust.

Turns out, while that's true, that misses the really amazingly messy reality of what it takes to do this. And one of the things that has been made more clear to me in the last three weeks that I've spent digging around in this is just exactly how much no one wants to do this, yet everyone has to. And I had to answer for myself the question, why doesn't Android? Because Android has none of this. None. Android does no certificate trust checking at all. It's just not in there. It's not in Linux, which is why it's not in Android. Linux falls back on OpenSSL, which does have all of this stuff in it. But the applications that use OpenSSL bring OpenSSL along with them, and so they get this trust chain checking. Android doesn't have it.

Similarly, Firefox has always done its own trust chain checking through, famously, its NSS, the Netscape Security Suite - and it's known by other names, I've seen it called other names - which provides that. And then they move to a different package which was translated from Java into C. And that's sort of a mess. And they're just now in the process of coming up with something that they call mozilla::pkix, which is their rewrite of this library that they're - I think it's slated for version 20, or, sorry, 31. And it's in the nightlies now, if you turn it on. And it's causing some trouble, but they're working it out.

So my point is that this is a big deal. And we've never talked before in all of our discussions about what a big deal it is. And talking about why the support for it is so spotty won't make any sense unless we understand why no one wants to do it, I mean, why it's a mess. So what's happened is, and this isn't a surprise to anyone who's been following these sorts of things, is the very first specification, this is the so-called X.509 spec, which is the standard for certificates, the designers created something that they thought would be sufficient.

And then needs changed and grew and evolved. And so things were glued onto that original spec. It was extended in various ways. And hackers found attacks against it that the original designers hadn't foreseen. So the original design wasn't all that it could have been. So that had to get patched up in order to defend against hacks. And so that's sort of been the history over several decades.

We have basically something that the people who are really in the know would just love to flush, I mean, just love to - they hate it. And they'd just like to scrap it and start over because, as is so often the case, we have a feeling now that we know how to do it right. It's not clear that we wouldn't just end up with the same. But what we have, unfortunately, can't be scrapped because it's the entire basis of the Internet's current Public Key Infrastructure, used for signing code, for signing drivers, for signing web servers, for verifying and validating all of this. This is a massive infrastructure that can't be changed.

So part of it is we need - we've evolved this over decades, and we've extended it in fits and starts. We still need backwards compatibility with everything that's come before because there's a whole bunch of legacy stuff running that, you know, we can't break those things. And we're still wanting it to do more. We're asking, because it's the only game in town, when a new need comes along, it's oh, look, we can - well, just like I'm talking about. We're talking about, like, OCSP Must-Staple. It's like, where shall we put that? Oh, let's put that in the certificate. Let's add one more thing to it.

And then people say, yeah, but that only protects the end cert. It doesn't protect all the certificates in the chain, and we really need that, too. So after the single extension is created, then it's like, oh, yeah, we didn't think about that. Okay. And so now we've got to fix that, too. So that's really, you know, where the rubber meets the road is very different from sort of the ivory tower description of how this works. Also remember that, in terms of the mechanics, we have a chain which is from the root out to the end certificate. Which is to say that the root signs the intermediate certificate, which signs the end certificate in the case of a three-cert chain. The linkages are by the issuer name, the issuer of the certificate. Their name will be the subject name of the certificate that signed this one. But there's no pointer to it. That is, we need to find it. We need to look it up. And, similarly, it will be signed by a certificate whose issuer name is in it. And then we need to find the issuer's named certificate which will have the subject name.

The point is the chain is actually going in the reverse direction from when we were at an end certificate. We need to search against the chain direction, which means every link we have to search our database, essentially, of certificates. Also there is the subject name/issuer name linkage. There's also something called the "subject key identifier" and "authority key identifier," which are the same things, but they're hashes rather than human-readable names. We decided at some point, oh, that's better because what about name collisions, which can be a problem.

So the other thing that this means is, if you are starting in a certificate, and you're searching for certificates that match the signer and so on, you may not have only one answer. You could have multiple certificates which match. And so you don't have a single path. You can have multiple paths. And in the mature logic, which has been worked out for this, there's something called "certificate validation path resolution," which is like a thing where, I mean, it's like a whole job just resolving the paths, the multiple possible paths back to a certificate you trust. And in doing this, you need to verify that the public key algorithm and the parameters of each certificate are checked and valid because, remember, this is a classic case of weakest link in the chain, literally a chain, and links. And bad guys will pry themselves in and figure out how to take advantage of any mistake made anywhere, in any of this.

So we have overly complicated, overloaded technology carrying age and evolution and backward compatibility, trying to be everything for everyone, that also has to be absolutely perfect, or any weakness will get found. We also have to check every certificate to make sure it hasn't been revoked. We need to know the current data and time and make sure that it is valid now and that the time is between the issue date and the expire date, so it's neither too early nor too late for us to trust this certificate.

There are other checks also because certificates have purposes. For example, some could be for authenticating a server, some for authenticating a client, some for securing email, some for signing code, some for timestamping, and combinations of any of those. So at every stage we need to look at the so-called "key usage extension" to make sure that the certificate states that it's valid. And remember that, since this is a chain, then we have to make sure that the things it's valid for, it covers all the other validity out to the end of the chain as we move back.

So then there's, like, path length extensions. We talked about that, I remember, once, where a certificate is able to assert that, from this point forward, you can only have one additional certificate. You cannot have two. That helps to prevent some sort of an exploit where you get a certificate mistakenly able to sign others. That has happened. And so, if you issued a certificate that was supposed to be an end certificate, but the flag was set saying that I can sign certificates, suddenly you've given somebody a certificate that essentially makes them a certificate authority, a trusted CA, because it's signed by someone that's trusted.

So again, the so-called "path length extension" says, from here, only allow exactly one additional certificate in the path from this point. So my point is that all of these things are there for a reason. They all impose constraints on the process in terms of length of the chain, the way we're going to chain, the validity, the revocation, the permissions allowed. And the nature of the trust chain is that every stage of the chain may restrict the rights further so you have to make sure that the rights you're wanting at the end of the chain are supported by every stage back. And even then you're not guaranteed to have a single chain. You could very easily have more than two, or more than one, two or three, and a couple of them may not be valid. So the moment you find one chain which is not valid for some reason, that doesn't mean that you need to fail the entire test. It just means you've got to make sure you're not able to somehow build a different chain which is valid.

So it is, unfortunately, just as a consequence of the way the technology has evolved, how much we're asking it to do, and how many exploits we've had to work against, it is incredibly difficult to decide, just to answer the simple question, can I trust this certificate, at this moment in time, for this purpose? I mean, that's the question we're wanting to answer. And, boy, is it not easy.

So that actually is why it isn't in Android today. It turns out that Android's Java interface for these functions doesn't even offer the features. It just isn't there. So for what it's worth, the only browser which bothers and is able to check revocation in Android is Firefox. Chrome can't. And in fact, even when Chrome had hardwired the revocation of GRC's revoked.grc.com site, immediately, across the industry, no one could get to the revoked site in Chrome, not because Chrome's revocation worked, but because they pushed out an update to their private list. But even on Android it couldn't block it because Android won't even give its client applications the access to the end certificate that the website is using.

So, I mean, it's very broken. We'll come to, like, at the end of this, do we care, because maybe we don't. But for what it's worth, because Firefox has always provided these features, they're the only ones who provide them on the Android platform today. iOS is only a little bit better. iOS checks revocation only for EV certificates. Now, clearly Apple has the technology to do this because, if they're doing it for any certificates, they could do them for all. So this is a tradeoff that Apple has deliberately made for overhead.

I've read many times in the last three weeks that this is not being done for mobile because of the overhead it represents. I really question that. That may have been true five years ago. I doubt that it's true today. At the same time, I did see tweets from people who turned on hard-fail option in Firefox that we talked about last week, turned on that second revocation test. And most people have found that it worked without trouble. But of all people, it turns out that Google sometimes fails the online certificate status protocol query that is being made by browsers.

So, again, this may just be that Google doesn't think OCSP has any value, so they're not putting any time behind it. But largely it's been successful. So I'm not sure that I buy this notion that there is, in today's world, still this tradeoff. And iOS could probably add this. At the same time, as we wrapped up last week saying, as soon as servers begin providing the OCSP status with the certificate, that is, have a certificate that's good for several years, and a fresh assertion, no more than a day old, from the certificate authority, saying yes, it's still good, when that's provided at once, then I think a lot of this is going to get fixed. We'll essentially be solving this problem of there being any overhead associated with making sure that certificates have not already been revoked.

Right now the mobile platforms are saying, oh, no, no, we can't do that. Then iOS is saying, well, okay, we will do it for EV certs. And one other reason is that part of the extended validation requirement is that OCSP be provided by the certificate authority. It's not the case that all certificate authorities have to provide the real-time online certificate status protocol. They could just be providing CRLSets. But if you're going to be creating and signing EV certs, you have to provide OCSP. So that may have also factored into Apple's logic in deciding, okay, on iOS, on iPhone and iPad, we will perform revocation checking for EV. Android doesn't even do that.

So Windows and Mac and Linux are variations on that. The good news is that, after IE7, revocation checking is on by default. I think it's 7. Might be 8. But it is on now by default. So IE and Firefox and Safari all have revocation checking on by default. Only Chrome has it disabled by default. In the case of Firefox, it's got its own library. So it's cross-platform. And I've ended up coming away very bullish about Firefox. The fact that they're able to turn on hard fail and succeed with that today demonstrates that they've got their code working, and that they are absolutely ready to respond to the OCSP Must-Staple initiative as soon as it happens, in a response header from the server, where the server can say, I am offering stapling, do not proceed without it. And eventually in the certificate, as soon as we - there is again, as I was saying, another enhancement to our certificate system. As soon as that's present, then the certificate will assert that must be provided.

So Firefox is really ready to make that move and will probably be able to do so quickly. It turns out that Windows has a little-known option since IE7, and it's in XP, which can be turned on to also enforce revocation checking. And on my page, under the Certificate Revocation site, I now have a number of pages. There is one that talks about browsers and OSes. So anybody who wants to experiment with this, I've got the links. There's just a simple registry entry that turns that on. It is disabled by default, and it's deeply buried down in the registry. So they didn't make it accessible at this point. But again, the fact that it can be turned on, and it doesn't break anything, demonstrates that Windows also is ready to go when that is enforced.

Unfortunately, I've heard from developers who have worked with the Mac. And Leo, your own experience and the experience of other listeners has been that requiring certificate revocation breaks all kinds of things in the Mac.

Leo: Yes. It's kind of unpredictable, too. It's, like, weird. It's like [murmuring].

Steve: Yeah. Unfortunately, it's just not ready. So I'm hoping that someone in Apple is seeing the writing on the wall that this is where the industry is going, and we'll be able to figure this - it was described to me as a race condition of some sort in the Mac where they just need to get it fixed. Right now it doesn't have any attention. I'm hoping that this whole issue will get additional attention.

And finally, the one thing - okay. So I guess the point I wanted to make was where is this being done? We talked about what a nightmare it is to do it. There are several libraries that have this. It is in OpenSSL. And if the Chrome guys end up deciding that they need to carry their own, then it will probably be - they'll probably look at OpenSSL. I heard, in some dialogue with one of them, I was told that, while, yes, OpenSSL had its problems, it's still better than everything else. So that's the feeling there. And, for example, if Android doesn't bring it, then if Chrome wants revocation, they're going to have to provide their own, which means you're going to have to have a mature library that is able to untangle this nest of certificate chain in the way that NSS does.

Mozilla is in the process, as I mentioned, of bringing a new library, this pkix library, online. And it's available in the nightly builds. Maybe it's enabled by default in the nightlies. I'm not sure where the status is because it does change from version to version. So Firefox has its own. Chrome currently seems to be a hybrid. I believe it uses something on Linux. And it may be NSS or some version of the Netscape Security Suite on Linux. But then both Windows and the Mac OS do provide this checking.

And arguably it ought to be in the OS. The reason it makes sense to have it in the OS is then all the apps running, not just a web browser, but other Internet apps, you know, once upon a time it was just email and web browsing and FTP that were Internet clients. Now, and certainly on mobile, we have this whole notion of an Internet application. Well, they're all depending upon security, all of those applications. And it makes no sense for them to redundantly each bring this nasty library along, which requires a team just to support this one function because it's such a mess. It makes much more sense for the operating system to provide those services.

Windows does. And actually Windows is very good and robust. The so-called Crypto API in Windows is there. It's serving their servers and the client platforms. The Mac has it, too, although they have a little work to do on the revocation side. iOS has it for EV, and I think they're just not bothering to do more because they don't have to. And Linux has none at all. It's not in Linux. Because, again, you really have to have a reason to need it. And no one's just ever needed it in Linux because the various - there's a few apps on the Linux platform, it not being mobile, bring their own in the form of typically OpenSSL.

So the one thing we haven't talked about, which is really interesting, is the way DNS can factor into this. Because think about two things. First is one of the great weaknesses of the PKI system, where we've got hundreds of certificate authorities that our operating system or browser, depending upon where that testing is being done, where that trust is being checked, hundreds of certificate authorities, any of whom can create a certificate for any domain, and our browsers or operating systems will trust them. My certs come from DigiCert. And our listeners know that. I know that. But the Hong Kong Post Office could sign a certificate for www.grc.com, and all the browsers that trust the certificates signed by the Hong Kong Post Office would trust that, even though it's absolutely not legitimate. But imagine if DNS was used to identify GRC's certificate authority. Then that solves, that eliminates a major category of potential problems.

And these are problems we've seen before. We've seen governmental agencies signing certs and using them for spoofing over in the Middle East in years past. And of course one wonders what our dear old NSA is doing, if they don't have a captive CA, or just tell a CA that they want a cert signed for the following domain. If DNS was used to publish the name of the valid signers for that domain's certificates, that's a huge win. And then you can go one step further.

And, by the way, there's something called a "CAA record" that is that. It's called the Certificate Authority Authorization. One way of using it is as I just said, for a browser to verify, by doing a DNS query, get the valid signer of the cert, and then check the actual cert it's given to see if that is the signer. That's one way of using it. The way it's been presented is to prevent a random employee in a company from getting an unauthorized certificate from some different certificate authority. So the idea is the certificate authority is supposed to check the domain for a CAA record, a Certificate Authority Authorization, to see whether they are the authorized signer for that domain's certificates, and refuse to sign a certificate for which they're not authorized.

Now, of course, a bad CA will not do that. If they know they're deliberately minting a bogus certificate, then they're not going to check that. But the idea was this would be a voluntary process that a certificate authority could use to prevent themselves from being spoofed by an employee of an organization not authorized to get certificates from them. So that's the CAA record.

There's also one called TLSA. That's used to associate an SSL or a TLS server certificate or public key, which are essentially synonymous, with the domain name where the record is found, that is, where this TLSA record is found, in order to form a TLSA certificate association, the idea being that, for example, I've got a certificate that GRC is issuing. I take the public key of that certificate and publish it in a TLSA DNS record. So I am saying that this is GRC's public key. Well, what that prevents is anyone else using a different certificate for GRC. Remember, there's absolutely no way to get the same public key because they have no way of knowing what our private key is. That's locked up and secret and never leaves the server. So there's no way for them to use the same public key as GRC without knowing the private key, which they can't know. So any certificate that was masquerading as a GRC cert would have to have a different public key. And if this was published in DNS, it can't - there's no way that anyone would trust that different key.

So to do this, though, we need DNSSEC. We need DNS Security. So this is all called DANE, D-A-N-E, DNS-based Authentication of Named Entities. And it's just another reason why the gradual movement of DNSSEC, DNS Security, these things just move slowly because this is a big infrastructure, and we need new versions of servers and clients and all the middle machines. We need to make sure we don't break things. And we need to make sure that the specifications work.

So it is a slow movement, but we're getting there. And when we have nonspoofable, nonforgeable DNS records, then that's really powerful. We've talked about all the obvious uses, just being able to be much more sure that we don't have a spoofed IP address when we're going to a site. That will be valuable. But then we can actually use DNS as an Internet-scale secure database and put all kinds of things in it in order to enhance security.

So lastly, does revocation checking even matter? We're arguably surviving without it so far. At the same time, anyone who has a certificate, like an owner of a certificate - for example, again, I'll use myself. I would - in no way would I want a GRC.com certificate loose in the world. I mean, just that would horrify me if there were such a thing. It's never happened, as far as I know. But if I have ever had a reason to believe that it had gotten loose, my god, I would want it revoked. I would want to absolutely foreclose any use of that certificate.

The problem is today I really can't. And the fact is, those 140,000 certificates that were revoked after Heartbleed, the revocation was pointless because, I mean, and this is where Adam is right in his argument. My issue is that he's not helping to move us forward. He's saying it's broken, so we're going to do our own thing called CRLSet, and they've been promoting it as useful. It's actually not. So my argument is let's fix this. Let's explain that this is not working. I mean, when I did this revoked.grc.com site, it generated a huge amount of concern because people just assumed it was working. It's not working. And I keep hearing from people, oh, there's really no demand for it. Well, you're not going to demand something that you don't know you don't have. If you think it's working, and you think you have it, it's like, well, why would you want it? You've already got it. Problem is, you don't.

Leo: So you and Adam agree on that, that revocation doesn't work.

Steve: Correct. Adam's point, and this is the key, is it's not that a revoked certificate might not be listed in the certificate revocation list. It probably, I mean, it absolutely will be. Certificate authorities are listing certificates in the certificate revocation lists, and they're publishing them in the OCSP protocol. Adam's point is, but those can be hacked, too. That is, yes, that they're not secure. So any bad guy who is really wanting to abuse a certificate would not simply set up a spoofed server, but would also arrange to defeat the revocation tests.

And so my argument is, well, it's possible that an attacker could also defeat the revocation tests. But it's also possible that they may not be able to. That is, the nature of the network may allow them to intercept the traffic for a website, but the traffic from the user to the revocation servers may not be accessible to them. So it's not the case that in all instances every single safeguard can be subverted. My feeling is that we know what a speed freak Chrome is. I mean, Chrome, that's what they sell is that Chrome is faster. One of the tricks is they absolutely don't check revocation. Because we do need, in order to do revocation, we do need to ask if the certificate we've just received is still good. It may be cached, in which case there's no delay at all. We may get a quick response, in which case it's negligible. We may be made to wait.

Google has said, philosophically, we think it's all garbage. We're not going to slow you down for something that could be broken by a determined attacker. My concern is, look, in doing that, we're not shining any light on this problem. We're not explaining that something that is potentially valuable to have, we don't have, because the companies who think their certs may be stolen absolutely want them revoked to prevent their services from being spoofed.

And so now we're going to see. Basically we've got this broken. Hopefully we're close to getting this thing fixed so that it cannot be - so that it won't slow things down, and so that it will be robust, and the potential for it being subverted by attackers will be eliminated. The question is, will there be an event which occurs before that happens, as a consequence of no one actually bothering to check for revocation? We're doing this lazy check, the so-called "soft fail," where if you hear nothing, you assume it's okay.

Now, turning this on in Firefox is extremely good. I mean, turning it on in Firefox defeats Adam Langley's argument completely. Now Firefox won't give you the page unless it can affirmatively verify that this certificate is not revoked. That's what that does. I've had it on always, and I've never had a problem. It's true that, if you're, for example, at Starbucks, and you have to log in, and you're doing a secure connection to the so-called portal, and it won't let you get to the revocation server, then that's going to be a problem. So you turn it off, you log in, then you turn it back on again. Or if you're at home, and your machine doesn't roam to Starbucks, turn it on. People have had it on now for weeks. And there've been a couple problems with using Google. But most people are reporting they're not having a problem.

So for me, here, I've got both checkboxes on in Firefox and, you know, because now I have actual nonspoofable hard-fail revocation which is working. Unfortunately, you can do that, well, you can do that on Firefox on the Mac. You cannot do that with Safari because to get that you need to turn it on in the Keychain, and that part is broken in Mac at the moment. Hope, well, they're going to have to get it fixed because they won't be able to support the OCSP Must-Staple, which is essentially hard fail, unless they get that working. So I'm presuming that they will.

Leo: So you and Adam agree also that stapling is the solution.

Steve: Yes. Yeah, I wouldn't say he agrees. He just - he's got a long history of sort of inertia. He has said, well, maybe it's the solution. I think it's clear that it's the solution. And everybody else understands that it is, and that's what they're working towards.

Leo: This guy's, who, a software engineer at Google? What does he...

Steve: Adam? Oh, yeah, I mean, he's been running the direct - he's been steering Chrome now for years. He's got - ImperialViolet is his blog.

Leo: Yeah, that's where I've read his posts.

Steve: Yeah. I mean, he's a...

Leo: He's a little snarky about you. I mean...

Steve: Oh, yeah, yeah.

Leo: Do you think that he fairly addresses the issue? I mean, his main point in his most recent post is that the revocation files would just be too big.

Steve: Well, which is making my point. I did a page where I went through and looked at all the available data. And so he's trying to demonstrate that their system can't work, that is, this CRL list can't work.

Leo: Because it's big, yeah.

Steve: Exactly. And my point was that, in fact it was Larry Seltzer in ZDNet did a column that said "Google does certificate revocation better." And it's like, oh, Larry. And I wrote to Larry. I've known him forever. And as you know, he's been in the business forever. And I said, "What? It's completely broken." And he sent something back that was sort of nonresponsive. And so I thought, well, okay. But he quoted a Google spokesperson saying, oh, yeah, when we're notified that a certificate is bad, we add it to the list, and users are protected. And it's like, that's a lie. I mean, it's an absolute lie.

And so I created a page which goes through and demonstrates that, based on the size of the CRLSet, which is what Google uses, that's the only revocation they use, they can maybe list 1% of the Internet's certificates, and they do list them from exactly 53 certificate authorities. Yet Windows trusts 353. So they've selected CAs. And any other certificate authority they blanket trust because they're not looking at the CRL lists from any of those. So they've basically chosen some little subset, strong on CA certificates and EV certificates, so that they're giving them priority. But that's a tiny minority of the certificates on the Internet. And...

Leo: Well, I guess Adam's saying that, but if we get the most important ones, that's sufficient.

Steve: Right. Well, that's what he's saying, and that isn't sufficient because, for example, 140,000 were just revoked by GlobalSign, and they have none of GlobalSign's certificates. Zero. That's not one of the certificate authorities in the list.

Leo: Certificates are frequently revoked for other reasons besides security; right? I mean, it's...

Steve: Well, no, 44% of them are - that's another thing he says that is not accurate - 44% are revoked because of key compromise. The majority reason is key compromise. I show a pie chart over on the OCSP Must-Staple page from either Websense or the 'Net, can't think of their name, 'Net monitoring guys. Anyway, I'm blanking. But, no, key compromise is 44% of all revocations. And all 140,000 that were revoked after Heartbleed were due to a concern over key compromise, none of which are handled by, none of which are seen by Chrome.

Leo: So his position, and I don't want to misparaphrase you, Adam, but his position sounds like it's too hard to do well, so we'll do the best we can, but you can't ask for any more than that. And yet you're saying, oh, but Firefox does do it properly. It doesn't get bogged down. It isn't giant - they're not giant lists. So are you saying at this point use Firefox?

Steve: I am, yeah. Or don't care. I mean, that's the other thing. We're...

Leo: Right now it's safe not to care. But you're waiting for the big event that might make it...

Steve: Well, it's safe until it's not.

Leo: Right.

Steve: It's safe until someone takes advantage of this. The whole industry knows this is a problem. I just want us to be honest. I just want to say, look, it is broken. Chrome is providing no protection. And here's my point. When I did revoked.grc.com, their system didn't pick it up. They added it to the header. They put it in the header of the...

Leo: That really is kind of a snotty thing to do. Well, we have to - we'll show Steve.

Steve: Exactly.

Leo: But it's not, obviously, that doesn't scale, either.

Steve: No. Exactly.

Leo: Get a hand out of reach of them.

Steve: There were three certificates there. Now there are five because they added cloudflarechallenge.com because that was known to have been revoked. And so they'd have egg on their face if they didn't show it revoked. And they added revoked.grc.com, mine. So they went from three to five. Yeah.

Leo: Okay. So...

Steve: That's the story, yeah. So Firefox is the safest because it brings its stuff along. It runs on all platforms. You can turn on hard fail and see how you feel. I mean, you get absolute security if you turn that on. It can false positive. Sometimes just trying again it will work, which is what the people that were having problems with Google found. Or you can not care. I wouldn't - I'm not arguing that everyone has to care because, I mean, Android has no protection at all. Maybe someone's going to take advantage of that, maybe not.

Leo: Would we know, I mean, would it be, like, such - we wouldn't because you could have a spearphishing attack that wouldn't be a widespread attack. They'd just take advantage of...

Steve: That's exactly it. The smart guy is not going to make a big splash because then Adam will immediately put that cert in his URL book.

Leo: He'll be in the header. Welcome to the header.

Steve: Yes. And welcome to the header. And so...

Leo: So if you're smart, and you want to, let's say, attack a bank, quietly do it and...

Steve: Exactly. You arrange to get their key, and you arrange to divert their traffic to your spoofed server, and you drain people's money very quietly and get as much as you can before the jig is up. And if we, now, if that were a revoked certificate, and the browser was checking, you wouldn't be able - they wouldn't be able to spoof the site. But as it is now, it's completely spoofable. Actually no one is checking.

Leo: Well, Adam, instead of resorting to snotty ad hominem snark, maybe you want to address this in a more appropriate way.

Steve: And I know they're going to. I mean, there are guys at Google who get it. There are guys who feel that OCSP Must-Staple is a direction they need to go in. I sort of think, from the thousand yards, I think Google tried to launch their own. They tried to go their own way. I've had some conversations with people in the certificate authority industry who haven't been happy with what Chrome has done. Their original, when they rolled this out, they wanted the CAs to submit to them some carefully curated CRLs that they would then amalgamate and use for Chrome. And the certificate authority community just blew them off. They said no, you know, we're doing what we're going. And so I actually think that the CRLSet is a failed solution to the recognition that the current system has problems. But it's the wrong solution, and we're moving toward the right solution.

Leo: Well, it's a great argument and a great conversation, and let's move towards the right solution and not just say, woo-hoo, hand-waving, it's not possible.

Steve: Well, and also saying, oh, don't worry, you're protected.

Leo: It's going to be fine.

Steve: That's wrong, too.

Leo: We've got an imaginary process that will protect you forever.

Steve: Exactly. Don't worry about it.

Leo: Don't worry. I'm going to start using Firefox. I'm a big Chrome advocate, but I think this is a good reason to start using Firefox, and I think everybody who listens to the show should start using Firefox. Firefox 30 is quite nice, actually.

Steve: Actually, they've really done a beautiful job of it, yeah.

Leo: And we've been critical of Firefox in the past. It's not like we play favorites here. One thing we all agree on, don't use Internet Explorer.

Steve: Yeah.

Leo: Steve does this show every Tuesday at 1:00 p.m. Pacific, 4:00 p.m. Eastern time, 20:00 UTC on TWiT.tv. You can watch live, or you can watch after the fact, on-demand audio and video always available. You've got 16Kb versions of the audio for the bandwidth impaired from Steve's site, GRC.com, along with full transcriptions, which actually is really great is you're having a hard time following. You can just read along as Steve talks. We have higher quality audio and full-quality video so you can stare at Steve waving his hands. I don't know why you'd want to, but you can.

Steve: And the blinky lights behind me.

Leo: The blinkin' lights at TWiT.tv/sn for Security Now!. Do visit GRC.com for SpinRite, the world's best hard drive and maintenance utility, maintenance and recovery utility, and of course all the great free stuff Steve offers all the time. And to follow along with SQRL. What are you going to do next week?

Steve: Next week we've got a Q&A.

Leo: All right.

Steve: So submit your questions, GRC.com/feedback. I'll go through it, and we'll have a great Q&A episode. And if all stays well, the week after we'll talk about the cool challenge of robustly harvesting entropy.

Leo: Ooh. I think that sounds really fun. By the way, Firefox 29, not 30. 29 just came out.

Steve: Yeah, 29.

Leo: All right, Steve. We'll see you next time on Security Now!.

Steve: Thanks, Leo.