How the Heartbleeds

Description: Leo and I discuss this long-anticipated, final "Second Tuesday of the Month" patch update for Windows XP - which has finally arrived. We share a bunch of interesting miscellany, then take a very deep dive to examine and understand the technology, events and implications of yesterday's (April 7, 2014) discovery of a two-year-old critical buffer overrun bug in the open source industry's OpenSSL protocol package. It's been named "Heartbleed" because it abuses the new TLS "heartbeat" extension to bleed the server of critical security information.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-450.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-450-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson's here. We'll commemorate the end of XP today. This is the last update for Windows XP. And we'll talk about a new exploit in OpenSSL that's got everybody shaking in the knees: Heartbleed, our topic, next on Security Now!.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 450, recorded April 8, 2014: How the Heartbleeds.

It's time for Security Now!, the show where we cover your security, your privacy online. And of course there was no better person when I first thought of doing this show than Steve Gibson to host it. He's the Explainer in Chief. He's got his finger on the pulse of security. He's an expert on technology. And he's here right now. There's his finger.

Steve Gibson: My finger on the pulse.

Leo: On the pulse, yeah. It is literally pulsing today. Oh, boy.

Steve: Oh, yes. So I was saying to you before we began, before you pressed - while you had your finger on the record button, that I was holding onto today's topic, the originally planned-for today's topic, that I've been working on for some time, which is harvesting entropy. We've talked all around the issue. But I'm now in the middle of it because I am working on SQRL's entropy harvester. And so I'm really tuned up for, like, exactly what are the challenges that a developer faces. And I thought this would just make a fabulous

podcast.

And so I was, until middle of the morning, I was holding onto it, knowing that we would have to talk about Heartbleed, but still planning to make it a bullet point at the beginning of the show. And finally that resolution just collapsed because just too much was happening. And I thought, okay, no. This is what the podcast is for, is for exactly this kind of thing. I'd be really annoyed if this happened on Wednesday because then we'd have to wait a week. But it gave us this fabulous opportunity last night. So today's podcast is titled "How the Heartbleeds."

Leo: And the Heartbleed exploit, of course, an OpenSSL exploit that's been in all the news today.

Steve: Yes. And of course much overhyped, as always. Much misunderstanding about what it is and why it is. Like, do we need to change our passwords? What does it mean, blah blah blah. Anyway, obviously, nowhere will you get more comprehensive coverage of it than here because I have spent the last 24 hours pretty much refocused on this as I kept getting pulled away from what we were planned to talk about, which we'll talk about in two weeks. And I'll actually have numbers and interesting additional data for that topic because I'll actually have already written the entropy harvester by then, so we'll have a different approach. But famously, I mean, as if there wasn't - actually this is sort of a perfect coincidence, that we also are - this is the long-awaited April 8th end-of-XP doomsday, XP-pocalypse and so forth. And nothing happened except that the open source system completely melted down, the non-Microsoft SSL library. And Microsoft of course had no problem with SSL at all.

So that's where we are. We're going to talk about this final Second Tuesday month finally arriving. I have a bunch of fun miscellaneous stuff, including a must-see documentary recommendation. And depending upon when you hear the podcast and which theater it's still in, you may be able to see it. I saw it yesterday and tweeted that it was the best $6 I had ever spent in my life. And what was funny was I was with an elderly neighbor who said that, exactly that phrase, as I was, like, writing that, getting ready to tweet the news of what I had found. And then a quick update on SQRL and SpinRite. And then we're going to dip into absolutely front-to-back coverage of what this OpenSSL vulnerability means to websites, to the industry, to end-users, and where it came from, and what it is, exactly.

Leo: Awesome, awesome, awesome. All right. Where do we start? It is D-Day for XP.

Steve: Yeah, it is. And everyone by now knows that I think it's going to be a big dud.

Leo: Just like Y2K.

Steve: Well, differently. I think Y2K didn't happen because people really did fix the machines which really were going to have a problem. And so I remember I was up at midnight, waiting to see if anything was going to happen, and nothing did. This is different because I'm not convinced that there is a problem. And it's even made mainstream news now. It's like, oh, my god, this is the end of Windows XP after 13 years

and blah blah blah. And my position is that there could be some unknown problems in XP.

But all of what we are seeing, with few exceptions, are problems in the apps, in the things that run on the operating system. That's what gets exploited. The browser gets exploited, Flash, Java. I mean, this podcast is all about those things. It's true that there are occasionally kernel OS-level problems. But, for example, even the topic of today's podcast is not about Apache or about Linux. It's about OpenSSL, a library running on top of the operating system. So my argument is nothing is going to happen, I mean, that all of the things that we run on XP, they continue to be patched except for Office 2003. That stops.

Leo: And IE. Got to remember, IE also is not going to be updated anymore. And in fact it hasn't been updated in a while on XP.

Steve: Right. And so, as we've said...

Leo: Don't use IE.

Steve: ...Chrome or Firefox, exactly. So anyway, I think this is a tempest in a teapot. I could be wrong, but I'm happy to plant my flag, and we'll all see whether I end up with egg on my face or not. I think, I mean, I'm going to keep using it. I haven't patched since '08, I think it was. And I'm just fine. And I'm not going to stop right now, working on SQRL and then on SpinRite 6.1, I'm not delaying those in order to move to Windows 7 because for me, it's a huge, I mean, it's weeks of downtime to set everything up again. And it is unfortunate that that's what Microsoft has done with their operating system, that you just can't move everything to a new platform, but you just can't. You have to reinstall everything.

So we'll see how this goes. There really is no news. You were showing a minute ago a clock that Microsoft has created. I mean, Microsoft is milking this for all they can in order to generate revenue, to force people off a platform which is working just fine. And you've probably seen in the news also that some major governments, I know over in the U.K. several large organizations have paid lots of money for the privilege of continuing to receive the XP patches which the rest of us won't be getting any longer. So Microsoft is now turning their vulnerabilities into a revenue stream for the first time. Which, I mean, remember that when the idea of getting auto-patched first happened, it was really controversial. We've gone from that extreme to exactly the opposite extreme, where people are freaked out now about this IV drip being turned off. So anyway, we live in interesting times.

Leo: Yeah. I don't think it's going to be a catastrophe. The ATM machines, as you've pointed out, Ed Bott tweeted this, as well, are not being used in any way that's risky, probably. Nobody's surfing the 'Net on them or getting email.

Steve: Right, right. And the other thing, too, is this clock expires at midnight, I guess. It looks like it has about 10 hours to go. So I don't know, that's not quite - oh, yeah, I guess that is midnight on the West Coast. So, but remember that we got patches today. And it's worth mentioning...

Leo: Were there XP patches?

Steve: Yes. Well, not XP. And again, this is typical. There was the long-awaited Office 2003 RTF patch. Remember, this was a zero-day flaw that was found. We talked about it weeks ago, and I recommended that Fixit which simply unregistered RTF from Office so that, if you received a document in mail, and Word tried to open it, it wouldn't execute code on your system. So that did get fixed officially. No longer do we need the Fixit. And all versions of Internet Explorer, 6 through 11, also got - those were the two critical vulnerabilities which were fixed. And then there were two others that were important. So it was one of our low patch months, just as we give up on Office 2003. And I've already talked about, if you need to use Office 2003, you can, for, like, editing your own book transcript. Your own book manuscript is not going to attack you.

And if for some reason you can't use a later version of Office, and you need to be opening documents that may come from malicious sources, or you just can't restrain yourself from clicking on links in email, then you want to use Libre Office or Open Office, which are being maintained currently after Microsoft has set Office 2003 adrift. So, but my point was we got all of our patches this month. So it's not actually until next month, May, when the second Tuesday of May occurs, and we don't get XP patched. All the other OSes get patched, and the people paying for the patches get their XPs patched. But the rest of us who are using XP still don't.

That's actually a month from now is the first actual event where something happens. Unless, and I could be completely wrong, some incredible zero-day exploit hits tomorrow, and because the bad guys know that Microsoft won't address it. And I really do wonder, frankly, if something really bad happened to XP, one third of the operating systems still on the 'Net, if they wouldn't step up and fix it anyway. So anyway, as I said, really, really interesting times. Perfect for us and the podcast.

And by the way, I've also seen, amid the coverage, lots of misreporting, where people say that the XP patches will no longer be available. And as if saying the past ones won't be available, which is completely wrong. You can install Windows 2000 today and get the latest service pack and all of the update that occurred afterwards. Those are all still there. So anyone who - oh, and you'll be able, if you had an XP retail box, you'll still be able to register it and bring it current after today. So it's the flow of updates that stops. But Microsoft's massive database of all prior fixes, that continues to be available, maybe forever. I don't know that they've ever - I guess you probably can't get them for NT 3.51. But I'm not even sure about that. I think they always have that available.

So the best $6 I ever spent in my life I spent yesterday seeing the documentary "Particle Fever," which is about an hour and a half long, about the Large Hadron Collider experiment.

Leo: Ooh, I want to see that.

Steve: And, Leo, it is still in theaters. My local theaters have it for, like, the rest of this week. So it got, on IMDB, an 8.1 out of 10 from 211 users. The critics' meta score is 87/100. The New York Times gave it 100. The Hollywood Reporter gave it 100. The Village Voice and Time Out New York both gave it 100, as did the Globe and Mail. And I'll share just three one-line summaries.

NPR said "It's jaw-droppingly cool stuff, explained with admirable clarity by an affable physicist tour guide, David Kaplan, and wedded to the tale of a massive technological undertaking like nothing in history." And they quote one scientist saying: "The biggest machine ever built by human beings." And NPR finished, saying: "And it's flat-out thrilling."

The Hollywood Reporter said: "'Particle Fever' succeeds on every level, but none more important than in making the normally intimidating and arcane world of genius-level physics at least conceptually comprehensible and even friendly to the lay viewer."

And lastly, Globe and Mail said: "Their excitement is infectious, and the entire endeavor is both mind-bending and tremendously human. Near the end, Peter Higgs, the recently Nobel Prize winner" - of course the Higgs-Boson is named after him, and this is what they were searching for - "and one of the scientists who first predicted the particle back in 1964, is seen in Switzerland watching the data results come in while a tear trickles down...."

Leo: Oh, I'd love to see that, yeah.

Steve: And Leo, I mean, I had tears in my eyes. This documentary, this was inside the project. When you look at the list of so-called "actors," it's all the physicists. And it says, "Played by himself," "Played by herself," "Played by himself," "Played by herself." I mean, the entire piece is, I mean, like Princeton physicists, Stanford physicists, Italians, Germans, Israelis, Iraqis. The point was made that this knows no national boundaries. Scientists from countries at war with each other are all there.

And so somebody was rolling a camera all through this, interviewing the physicists about what this means. So basically for an hour and a half we follow through the dream and the construction and those first stumbles that some of us will remember where there was a leak and a catastrophe. And then, like, they're arguing about whether to tell the truth to the media about when they are going to turn it on or not because they're so worried they're going to stumble, and to have the world's cameras watching them stumble. And so it's like the - it's pure science.

And then there's the other distinction made between the theorists and the experimentalists. And that line is drawn clearly, and you hear them each talking about each other. Oh, it's just - I cannot overstate how incredible it is. There is a site, ParticleFever.com, where thanks to Simon Zerafa, who shot that to me this morning, where you can go and see the couple-minutes-long trailer for it, if you can still find it. There is a list there of the theaters that have it, and it's in the major cities around. If it's near you, oh, my god, it's worth it not to wait till the summer. It will be out in HD and on disk this summer. You want to run this, Leo?

Leo: Should I run the audio for it?

Steve: Yeah, yeah.

Leo: All right.

[Clip]

Leo: Do they talk about next year's experiment? Because that's interesting, too.

[Clip]

Leo: Wow, that's a nice shot.

[Clip]

Leo: That's Higgs.

[Clip]

Leo: Wow.

[Clip]

Leo: I can't wait to see this.

[Clip]

Leo: "Particle Fever." Wow.

Steve: Yeah.

Leo: But it's not over for the LHC. They've got much more to do. We were talking with Michio Kaku about the next experiment, which may be just as significant. It's really remarkable. It's really exciting.

Steve: Yup, yup. So we'll definitely have this available this summer. But if it's in a theater near you, I mean, I was - I'm not sure yet that I won't be going there because I think I have to see this thing. I think I have to stand in it and look around.

Leo: Yeah, you know? We should do a field trip.

Steve: I was going to say that...

Leo: Let's do a TWiT field trip.

Steve: ...planned ahead, we would announce to everybody when we're going to be there, and anyone who wants to join us would be welcome to because I can't imagine a better place to have a little informal get-together.

Leo: It's actually showing just down the road apiece.

Steve: Oh, Leo. I'm not kidding.

Leo: I'm going to go see it, yeah.

Steve: I'm not kidding. And it may disappear on Friday.

Leo: Well, it comes Friday. So we're probably getting your old print. And it leaves Sunday. So it's a three-day engagement in San Rafael. So, yeah. But this is all on the website, ParticleFever.com. So you can see where it is. And of course we'll be able to see it someday.

Steve: Yeah. This summer.

Leo: Home systems.

Steve: The site says that it'll be available. So no one will have - and I will certainly let everyone know when that happens. But, wow, it's just, oh. It was, I mean, what was really interesting is that, as somebody who's been involved in creating stuff - and, for example, I no longer tell - I no longer try to guess when something's finished. Someone says, "When's this going to be done?" I have no idea. So I could relate so well to the dilemma that the scientists were in with $8 billion spent and this incredible worldwide energy, and then the press all, you know, not understanding at all that this is not like plugging in your coffee pot and it brews, I mean, no one has ever built one of these before. What's going to happen when we turn it on?

And so they made the greatest - they did a perfect job of illustrating that tension that exists between real experimental theoretical physics out on the far edge and the lack of understanding that anyone who isn't in that mode of going where we've never gone before, you don't know what is ahead. Which I argue is what makes the journey so fun.

Leo: Yeah, right.

Steve: But it annoys the media because they don't know how to reduce it to a sound bite.

Leo: Right. Can't win.

Steve: Yeah. So, yeah. "Particle Fever." If you can see it, see it. And I already mentioned, we talked about "Rogue Code," Mark Russinovich's book. I got it last week, and I'm about 16 or 17% in. And it's exactly what we got before - new topic, new stuff, real interesting. And, boy, I don't know how he's done it, but he's got some stuff in here about high-frequency trading that is like in the news yesterday. It was like, wait a minute. What? So it's also very timely. So this is his third novel following "Trojan Horse" and "Zero Day." And as I said, it's - oh, and Jeff Aiken, the character who he's developed in the two prior books, is with us again and is sort of the focal point of the novel. So I still don't know what happens, but I'm definitely enjoying the read. So he did say that it was late May, so late next month was when it was going to be released. That must mean hard copy because Amazon has that for preorder, but you can get the Kindle version now. And of course the Audible version is no doubt going to happen before long.

Okay, now, this I feel a little bit weird about. But I just want to give an acknowledgment to a very useful utility. I think I've mentioned these guys before. There's a product called AnyDVD produced by a company called SlySoft.com.

Leo: Oh, yeah. We've talked about them a lot, yeah.

Steve: And they have a Wikipedia entry. AnyDVD, under Wikipedia, says: "AnyDVD is a Microsoft Windows driver allowing decryption of DVDs on the fly, as well as targeted removal of copy preventions and user operation prohibitions (UOPs)." That's where, like, you can't fast-forward through the endless commercials that you are now forced to watch on a disk that you buy, for example. You can tell from my voice how I feel about that. "With an upgrade," says Wikipedia, "it will also do the same for HD DVD" - those are the red ones that were of course the ones I adopted before the world went to blue.

Leo: I should rip mine. If I had a player, I would. I don't have any way to play it.

Steve: Yup, "...and Blu-ray discs."

Leo: I have a ton of those HD DVDs.

Steve: "The AnyDVD program runs in the background, making discs unrestricted and region-free. In addition to removing digital restrictions, AnyDVD will also defeat Macrovision analog copy prevention. AnyDVD will not work on VHS tapes," they note, "only discs. Analog prevention distorts the video signal to prevent high-quality copying of the output. AnyDVD is also able to remove copy-prevention from audio CDs."

This came to my attention, I mean, first of all, I've known about AnyDVD for some time. Everyone knows I buy disks. What really annoys me is I'm supporting the industry which is using some laws of questionable value to attack other companies, which is annoying. But I buy disks. But we've also - I was comfortable with the idea that when you purchased the content, you had the right to use it as you saw fit, not to give it to people, not to copy it so that it would be hurting the revenue of the people selling it. But if I want

a disk to be viewed on my iPad, the argument is, that's within fair use.

And so anyway, these guys, of course, the movie industry has been after them forever. And they were sued in, I think it's Antigua. And they're appealing the decision. There was a $30,000 judgment against them, which is probably pocket change for these guys. Anyway, I just sort of wanted to say, you know, using this kind of tool in a way that doesn't reduce the revenue of the copyright holders, I think this is valuable because I'm annoyed when I can't use the chapter jump button in order to skip over an amazing amount of previews that I don't care about on a disk which I have purchased. So again, I just wanted to - for those who don't know, now everybody does, and I have said my piece.

This next picture, Leo, if you want to put this on the screen, I took some heat over in the GRC newsgroup over my calling Windows XP a "robust operating system" because some people who'd been around for years remembered when I was calling XP a "toy." And it was when we moved from Windows 2000 to XP that I just looked at it, I mean, I think I used words like "Romper Room" and so forth. Anyway...

Leo: Yeah, I think we called it the "Fisher-Price Interface" on TechTV.

Steve: Oh, god. Anyway, so what this...

Leo: Glowing buttons, you know.

Steve: Yes, yes. Well, and here we have this, I mean, we all survived Clippy, the jumping dancing paperclip from Microsoft Office.

Leo: The search dog was so much worse.

Steve: Yes. And so here is this canine that jumps up. And, I mean, in the menu of options it has "Ask him to do a trick." And, I mean, okay. So I rest my case. Oh, my lord.

Leo: It was a toy operating system.

Steve: What happened was I fired up a VMware virtual machine running an old, I mean, a virgin version of XP because I was curious to see when the GDI+ interface or API was added to XP. Was it from the beginning, or did you have to have .NET something installed? Because I couldn't get a straight answer. And so I went to search for that DLL, gdiplus.dll, and up comes this dog. And I looked, and I thought, okay. I had so forgotten about this and how an untamed Windows XP looked 13 years ago. So, yes. And so anyway, my point was I was able to provide this to the guys in the newsgroup and say, okay, tell me this is not a toy. Yes, you could turn it off. Yes, this operating system has had 13 years to mature. Yes, its firewall is on now by default. So it's as good and safe as it could be for what it is. But, boy, yeah, when the dog was jumping around and doing tricks, it's like, okay.

Leo: Well, and of course you could go to the classic interface, and you could turn off the dog even though it was kind of depressing because he'd hang his head and kind of mope off into the distance.

Steve: Oh, and then wanders off into the distance, sort of like, oh.

Leo: Yeah. Like it makes you feel bad for turning him off. Really? Really?

Steve: I know. I also have been meaning for weeks to note to people who I guess maybe thought I didn't know. When I was talking about The New York Times revelation about the NSA's breaching Huawei's network and products, they said - I got a lot of tweets and email from people saying, Steve, Ed Snowden is no longer in charge of disseminating this. He's not disseminated anything since he left for Russia. And so, yes, I understand that. My argument was meant to be, not that specifically. And by the way, The New York Times even said that these were documents not from Snowden but from other sources, although in the same story they sort of oddly referred to him and the NSA. So it was a little muddy.

But anyway, I had been meaning to acknowledge that I understood that. And my annoyance was that, when we most recently saw Snowden, his newest argument supporting his decision was Fourth Amendment, and that this didn't fall under that, and that arguably, if these documents did somehow come from him, he did have some responsibility to look through them and decide what he was giving to other parties. So I didn't want to bring it all up again, but I just did want to acknowledge it because I hadn't before.

Also our great listeners found for me a link between the Windows platform and iOS and the iCloud. There's something called PushBullet that Don Houle, H-o-u-l-e, tweeted me - as did several other people, but Don was first - which allows you to send stuff back and forth between your Windows device, I mean, even like there's a plug-in for Chrome on Windows that allows you to instantly shoot things from Chrome on Windows over into your iCloud-connected devices. So I wanted to notify anybody else who has the same problem. Something called PushBullet, and you just go PushBullet.com, and you can find it. And it seems to be - the guy seemed to be related to Google somehow.

And Leo, I wanted to mention that another Kickstarter success has occurred. I mentioned to you before, but here it is. It's arrived. This is the cold, the slow drip cold brewing project which I mentioned before. And it came a couple days ago. I haven't yet even taken the lid off. It's all still in here.

Leo: You know, didn't we try all this before? Coffee without a bite? You don't like it, Steve?

Steve: I didn't try it.

Leo: Yeah, yeah. You said, "I like a bite in my coffee."

Steve: Yeah. So we'll see.

Leo: I know why. Because you tried the black blood of the earth, or the dark blood of the earth, which is a cold...

Steve: You're right, I did.

Leo: Yeah, it's a cold brew, and that's why it's low acidity. We'll see. Maybe you'll like this. It's pretty.

Steve: Yeah. I just haven't, you know, I just want to set it up and watch it drip. And I'll probably stay with the solution I have, which is generates a pot that I then drink over the course of a couple hours. And everyone who tries mine, loves it.

So, SQRL. We have two additional languages that are now supported at request from our listeners: Lithuanian and Latvian. And we are now at 54 languages, ready to be translated as soon as I get the user-interface text all finalized. And we're up to 362 participants. So we will have - it'll be great once we're able to turn everybody loose. As I mentioned last week when I talked about the UI engine that I had written to be multilingual and also to size itself with a real focus on optimizing per-language the metrics, aspect ratio and so forth, of the user interface, that's done.

So the first thing I'm working on now is the harvesting of entropy from the users' system on a Windows platform so that we're able to securely generate identities that are richly 256 bits of entropy for the ID and also for other parts of the crypto system that need it. So I'm working on that now. I had planned to talk about that today. Instead, we'll talk about it in two weeks. And you'll see that the tense of what I'm speaking changes because instead of "I'm going to," it will be "I did," and I'll also have results to share, which will be cool, like the rate at which entropy is available from different sources I should know two weeks from now, after next week's Q&A. We'll talk about it.

And very quickly, for SpinRite, I wanted to share this one picture which was tweeted to me. And as they say, a picture is worth a thousand words. This shows - this was from Sean McCormack, who sent me a tweet with this photo just saying "Thank you @SGgrc," showing SpinRite, oh, not quite halfway through his drive. And I'm looking at one, two, three, four, five, six, seven, eight, nine, 10, looks like 11 or 12 green R's on the map which is half done. And green R of course means that SpinRite found a problem in the sector of his hard drive and was able to, through whatever manipulations were required, maybe invoking DynaStat with data recovery or not, was able to perfectly recover 100% of the data in that sector. And I don't know how the rest of the drive looked, but this is the kind of thing where you run it a second time, and then it's perfect. And the reason is that the problem sectors are rewritten perfectly, so that when the drive comes back along, there's nothing wrong.

So this is the kind of - you want to run your drive on SpinRite at this stage, when you're still getting green R's, rather than waiting until it's too late, where you get red U's, because then while SpinRite is still able to perform a partial recovery, and that makes it entirely unique in the industry, you may be missing some bits that are important. Or, as we've found, it might be a chunk of a directory tree, and you can still get to the rest of your drive which you were cut off from before, even if there are some bits that were not recoverable. So even partial recovery is better than none because, after all, there are,

what, 4,096 bits in a sector. But typically, if you can't read 11 or 12, everything else gives up on all 4,096 of them, which I never really understood. So anyway, another success from SpinRite.

Leo: Nice job.

Steve: Okay. So, "How the Heartbleeds." Two years ago, an RFC, I have it in my notes somewhere, 6250 something, just from memory, I don't remember our exact number, an RFC was finalized - I think it was in February of 2012, so just over two years ago - which added a new feature, which in the parlance of SSL and TLS we call "extensions." So this is a protocol extension to TLS, the Transport Layer Security, to create a heartbeat. And I suspect that this was actually done, not so much for TLS, but for DTLS. And I don't think we've explicitly talked about DTLS. That's the UDP version of TLS, which is to say, normally the way - and remember, I'm going to say TLS because this is all about TLS, but this is OpenSSL, this is HTTPS and so forth. I imagine our listeners all understand this.

OpenSSL went from v1 to v3.0, and then it was sort of phased over. The name essentially was changed to TLS 1.0, which is effectively the same as SSL 3.0. But at that point SSL stopped being the term, and TLS is now the term as we move forward. And we are now at TLS 1.2. So one of the - so normally the way we establish a TLS connection is we first establish an underlying TCP connection between two endpoints. Normally the client connects to the server. And then on top of that connection - so we call that the "transport layer." Then the application layer is data running on top of the data-carrying connection. So TLS runs on a TCP connection, sort of bound to it.

Well, it's also possible to - and above that is sort of like the real application layer, HTTP, which in the SSL version, the secure version, is HTTPS. And you can have - we've talked about secure email or FTP secure and so forth, different application layers running on the underlying protocol that underneath it has TCP. Well, the architects of these protocols decided there would be value in allowing UDP, that is, not TCP, but UDP, to be the underlying transport protocol and then to support TLS on UDP.

So what's different about that is there isn't the initial TCP handshake. And with TCP, there is the notion of a keep-alive. When users at home behind a NAT router are connecting to a server on the Internet, their outgoing TCP SYN packet, that is, TCP SYN packet, it creates a mapping through the NAT router so that the returning packet knows where to go inside the private network. So it actually creates a - you can think of it sort of like a temporary routing rule or a firewall rule that routes that returning packet to the proper server. That rule stays alive normally until the TCP connection is torn down. And that's done by either sending a FIN, an F-I-N, a finish packet, or an RST, a reset packet. Both of those are types of TCP packets, both either sort of gracefully or immediately terminating the TCP connection. And that's what tells the router that it can now remove the entry from its table.

Well, UDP doesn't have anything like that kind of protocol. It's normally used, as we've talked about, for DNS. You send a DNS query off, and a DNS response comes back. So the router creates a mapping that is relatively short-lived to allow the answering UDP packet to get back to its sender. The problem is nowhere in UDP is there a, like, we're done with this. That just doesn't exist. Where it explicitly exists with TCP, it doesn't with UDP. So, and I just gave the example of a consumer router.

But increasingly through the Internet is all kinds of state being maintained. You might

have enterprise firewalls which are stateful firewalls, allowing the users inside the enterprise's Intranet to have access to the Internet outside and monitoring that traffic as it comes back. So, and you can have proxy servers and all kinds of stuff now sprinkled around the Internet that is paying attention to state, not just passively moving packets back and forth. So it's probably the case that the guys who were doing the TLS over UDP said, you know, we need the equivalent of a keep-alive.

What I didn't mention on TCP is there is this concept of what's called a keep-alive, which is like a heartbeat. The idea is that either end is able to send the other an ACK packet which is just slightly incorrect. It's like it's acknowledging the past of their connection. And that induces the TCP stack on the other end to say, no, this is where we are in our agreed-upon sequence numbering of bytes moving between endpoints. So it's sort of a way to just - there's a way to poke the other side and say let me hear from you. And so TCP keep-alives are used on otherwise static connections. You don't need data to be actively moving through a TCP connection in order for it to be kept up.

Now, there are, because you can have, like, computers could just go off the Internet and leave state in place, that is, they may not send the FIN or the reset packet. They may never get around to it. So you still have timeouts on even TCP state in NAT routers and other places, where if some length of time goes by with no data passing, the router says, well, even though we never got the official word that this connection is no longer valid, apparently it's not because nothing is happening. So they'll tear it down. So the keep-alive is an affirmative way on TCP of just having a very tiny, because an ACK packet is very short, a very tiny, "Is this where we are?" And the other side says, "Well, no, we're here." And then it's like, okay, fine. And that keeps everybody happy.

So since UDP has no notion of that, this notion of a heartbeat was added, not to UDP, but to TLS, which is sort of this level above the lower-level transport. And it was added as an extension so that the endpoints advertise what extensions they support, and it would then be possible to achieve the same thing with a TLS connection over UDP as we have with the so-called keep-alives on TCP, and that is a heartbeat. And that's where the name came from, the idea being, even though no data, no TLS data is being exchanged, we want to continue to assert that at each end we've got a TLS stack, and we're both paying attention. And so this heartbeat, which was introduced with an RFC two years ago, is the way we do it. So one month later, after the RFC was finalized, in March of 2012, OpenSSL went from v1.0.0 to 1.0.1 and added this extension. And unfortunately, there was a problem with it two years ago.

In their security advisory dated the 7th of April, that's yesterday, OpenSSL said: "TLS heartbeat read overrun," and then they gave the common vulnerability extension, the standard CVE number. They said: "A missing bounds check in the handling of the TLS heartbeat extension can be used to reveal up to 64k of memory to a connected client or server. Only 1.0.1 and 1.0.2-beta releases of OpenSSL are affected, including 1.0.1f," which is the last one before "g," which happened yesterday, "and 1.0.2-beta1." They were getting ready to do a 1.0.2 release, which is in beta. It naturally had all of the current code from 1.0.1, which until yesterday all had this problem. And then they give thanks to Neel Mehta of Google Security for uncovering this bug, and our friend Adam Langley and some guy, Bodo Moeller at ACM.org, for preparing the fix.

They say: "Affected users should upgrade to OpenSSL 1.0.1g. Users unable to immediately upgrade can alternatively recompile OpenSSL with the -DOPENSSL_NO_HEARTBEATS option." So that's a compile time option that's always been available for the last two years. And they say 1.0.2 will be fixed in the 1.0.2 second beta.

So what does this mean? This means that, as early as a full two years ago, any websites

that were keeping up with the latest and greatest and therefore incorporated the then-1.0.1 version of OpenSSL have from then or from whenever they did until yesterday, and hopefully not today or tomorrow because this news was flashed like wildfire, for up to this two years and hopefully ending or ended now, there has been a vulnerability present which the Internet community at large has been completely unaware of which allows an attacker to exfiltrate up to 24K of memory that was meant to be private. Memory in the server - and it is a bidirectional exploit. So if the client had this, then something you connected to could come and get memory from you, as well.

But that's not where we're focused. We're focused on the server because the guys who found the problem attacked themselves. They attacked their own servers and saw what this 64K block contained. It's going to vary from attempt to attempt. There's no control over it. Like a buffer overflow, you get what you get. You get what's there. But unfortunately, 64K is a lot of memory. And they found their server certificate in that 64K and other critical, crucial private information, passwords and so forth.

And in fact Dan Goodin of Ars Technica has some nice coverage, which I'll share in a second. But the Tor Project that is of course concerned about this, they're systems are largely based on OpenSSL Protocol as a wrapper for what they do. They immediately blogged about this last night. And they said: "A new OpenSSL vulnerability on 1.0.1 through 1.0.1f is out today, which can be used to reveal memory to a connected client or server. If you're using an older OpenSSL version, you're safe," meaning if you never went to 1.0.1. Even recently, 0.9.8 had been, like, still maintained in parallel. So the v0.9.8 track was being kept up in parallel with the v1.0.1 and so forth track. So many people may have stayed safe. And in fact we have some numbers about that that I'll share because it's not as bad as the 66% of the Internet that the early reports have said. It turns out that many fewer servers, for whatever reason, probably because they weren't using the latest version, did have this extension exposed and therefore vulnerable.

The Tor blog continues: "Note that this bug affects way more programs than just Tor. Expect everybody who runs an HTTPS web server to be scrambling today." And we'll talk about LastPass because they did scramble already and have a great blog out about the consequences to them. "If you need strong anonymity," says Tor, "or privacy on the Internet, you might want to stay away from the Internet entirely for the next few days while things settle."

Leo: Wow.

Steve: And sadly, yes, sadly, that's good advice. And I'll explain why. Because note that this - and we'll come back to it. But this is not like the typical website loses 100,000 usernames and passwords, where suddenly, without you doing anything, you're now vulnerable. Remember that this is a server or servers. Credentials may have been taken. Which means, first of all, if they weren't using Perfect Forward Secrecy, and someone had archives of their traffic, then that past traffic could be decrypted. We've covered that in detail in previous podcasts. So there's one problem.

But maybe the more relevant problem, assuming you don't have archives of the past, is that with a server's credentials you could impersonate the server. You could do a man-in-the-middle attack; or, if you could, for example, somehow get the user to go to the wrong IP, like by changing their hosts file in their machine, or by somehow poisoning their access to DNS - we've seen routers, home routers whose DNS has been repointed - that would redirect them to a fake DNS server so that their browser would think they

were at Amazon.com, but in fact they're somewhere entirely different. Yet, with a spoofed certificate, with a valid certificate for Amazon.com, their browser wouldn't know any better.

So the point is that it's connections we make now, connections we make to sites that may have been compromised going forward, that we need to worry about. And so that also tells us that proper remediation for this is revocation of any - the revocation of the certificates of any site that may have had its certificates stolen during this two-year window, or however long the window was, because we need the old certificate to be revoked for use in our browsers. And that requires that browsers check revocation, which most don't by default. So we'll talk about that, too.

And then those sites need to have new certificates reissued. So one good sign would be to see whether the certificate on a secure site that you've not yet logged into, not yet given your credentials to, has recently been reissued. For example, if you check LastPass.com, you'll find that their cert says April 7th or 8th because they've just got a new one in order to address this. So that's one good thing we'd like to see. So let me go on with the Tor perspective because it's real-world and really useful, as you can see.

So they said: "Stay away from the Internet entirely for the next few days while things settle. Here are our thoughts on what Tor components are affected." And so they said, for example, the clients. "The Tor browser should not be affected since it uses libnss." Now, that's the Netscape security suite, which is not OpenSSL, so not a problem. And so the Tor browser isn't an OpenSSL user. But, they said: "But Tor clients could possibly be induced to send sensitive information like 'what sites you visited in this session' to your entry guards," says Tor. "If you're using TBB, we'll have new bundles out shortly; if you're using your operating system's Tor package, you should get a new OpenSSL package and then be sure to manually restart your Tor."

Then they said, of Tor's relays and bridges: "Tor relays and bridges could maybe be made to leak their medium-term onion keys, which are rotated once a week, or their long-term relay identity keys. An attacker who has your relay identity key can publish a new relay descriptor indicating that you're at a new location." And this goes on and on. If anyone's interested, I've got links. In fact, today's show notes is a link fiesta. So lots of things for further research of anyone who's interested.

So Tor goes on about their hidden services: "Tor hidden services might leak their long-term hidden service identity keys to their guard relays. Like the last big OpenSSL bug, this shouldn't allow an attacker to identify the location of the hidden service. But an attacker who knows the hidden service identity key can impersonate the hidden service." So in the case of Tor, the impact for them is complex because of the nature of the way they're using the OpenSSL package.

Dan Goodin, writing for Ars Technica today, did some nice reporting. He said: "Researchers have discovered an extremely critical defect in the cryptographic software library an estimated two-thirds of web servers use to identify themselves to end users and prevent the eavesdropping of passwords, banking credentials, and other sensitive data." And so he says the OpenSSL is the "default package used by Apache, as well as just about everything else that's not Microsoft-based." And that's exactly right.

So consider also that OpenSSL is what everyone in the open software world uses. I mean, almost without exception. We talked about GnuTLS a couple weeks ago, which a few people use, because it had some recent problems, too. But mostly OpenSSL is just the de facto default standard. And so chat clients, instant messaging, like everything that wants to connect securely that has its roots in open source and is establishing this kind of

a connection, probably has OpenSSL and probably has something from the last two years in it. So this has repercussions way beyond just web servers. And in fact we could argue that those issues may be, in the long term, more critical because there aren't sort of the single points of contact and focus that at least we have with web servers.

Dan continues, saying: "The bug, which has resided in production versions of OpenSSL for more than two years, could make it possible for people to recover the private encryption key at the heart of the digital certificates used to authenticate Internet servers and to encrypt data traveling between them and end users. Attacks leave no traces in server logs," which is absolutely true, "so there's no way of knowing if the bug has been actively exploited." So it's not possible to go back and check old logs to find this. Many times it's possible to do that, in which case companies are able to rule out a now-known attack having been used previously, before it was widely known. Not in this case.

Dan continues, saying: "Still, the risk is extraordinary, given the ability to disclose keys, passwords, and other credentials that could be used in future compromises. The researchers, who work at Google and software security firm Codenomicon, said even after vulnerable websites install the OpenSSL patch, they may still remain vulnerable to attacks. The risk stems from the possibility that attackers already exploited the vulnerability to recover the private key of the digital certificate, passwords used to administer the sites, or authentication cookies and similar credentials used to validate users to restricted parts of a website."

So just breaking from that for a minute, remember, this is just a 64K gift. This is a 64K, we don't know what we just got, but let's see what it looks like. Let's see what's here. Let's look for text strings. Let's look for recognizable x.509 certificate headers and so forth. A lot of this material now fits very well established standards, which means it's easy to scan through a block of seemingly random gibberish and find literally keys to the kingdom.

So Dan says: "Fully recovering from the two-year-long vulnerability may also require revoking any exposed keys" - I would argue that's the case - "reissuing new keys, and invalidating all session keys and session cookies." And then, finishing up, he says: "OpenSSL is by far the Internet's most popular open-source cryptographic library and TLS implementation. It is the default encryption engine for Apache," also the "engine-x" server, which by the way is spelled n-g-i-n-x, which is otherwise unpronounceable, but it's pronounced engine-x.

Leo: It's what we use, yeah.

Steve: Yes. It's a fabulous server.

Leo: Oh, yeah. It's the one, yeah.

Steve: "Which according to Netcraft runs 66% of websites." Now, this is where the 66 or the two-thirds is coming from. And this, as we'll see in a second, is a bit of a misnomer. "OpenSSL also ships in a wide variety of operating systems and applications, including the Debian Wheezy, Ubuntu" - by the way, I've been mispronouncing it "Ubuntu," and I got corrected by someone in Twitter, so thank you for that - "Ubuntu, CENTOS, Fedora, OpenBSD, FreeBSD, and OpenSUSE distributions of Linux.

Leo: Oh, I'm sorry, Steve. That's pronounced "SU-SE."

Steve: SUSE, okay. Thank you.

Leo: Please get it right. Thank you.

Steve: Thank you. "The missing bounds check in the handling of the Transport Layer Security heartbeat extension affects OpenSSL," and then Dan explains, "1.0.1 through 1.0.1f," as I talked about. And then he says here in his article what I already mentioned. So can it be exploited? The researchers who discovered it wrote, and I'm quoting now:

"We attacked ourselves from outside, without leaving a trace. Without using any privileged information or credentials, we were able to steal from ourselves the secret keys used for our SSL certificates, usernames and passwords, instant messages, emails, and business-critical documents and communication." So, and then Dan wraps up, saying: "They called on white-hat hackers to set up honeypots of vulnerable TLS servers designed to entrap attackers in an attempt to see if the bug is being actively exploited in the wild." And as I mentioned before, Heartbeat can be disabled through a recompile. But if you're going to do that, you might as well just get 1.0.1g and be current.

There is a great page that summarizes this. And this was put up by these guys. I don't know if it's going to be maintained over time. But it's just Heartbleed.com. And so there's a lot of good information there. Yesterday evening I immediately fired up a dialogue with Ivan Ristic, the guy who created SSL Labs, saying, "Ivan, I'm sure I don't have to urge you to add detection of this vulnerability to your service. But, boy, it would certainly be useful." He apparently did an all-nighter and shot me email in the morning saying, "It's up."

So the good news is the SSLLabs.com site that we often recommend and use is online, updated, and there's not a separate test. You just do the "random test a given server" and put the domain name in. And at the top of the reports page - and Leo, if you scroll down to the bottom of my notes, I took a screenshot of it actually for GRC. We were never vulnerable because we're using IIS. And I don't believe that extension is supported under Windows. But there is a new bar running across immediately underneath the big grade that you're given. Yahoo!, for example, was vulnerable. Twitter was reported as vulnerable.

So there have been a number of major websites that, not surprisingly, are running Apache or nginx with OpenSSL, and they were doing what they should have been doing, which was keeping current. And so for as many as two years, if anybody else knew about this, certificates and 64K of potentially value data could have been exported. Oh, and now I'm seeing in my notes the TLS Heartbeat RFC was 6520. And, yes, it was February of 2012.

Leo: There's a site, Filippo.io, that claims to have a heartbeat test, as well.

Steve: And, unfortunately, it doesn't work.

Leo: It doesn't work, okay.

Steve: It false positives and false negatives. For example, put GRC.com in right now. An hour ago it said I was vulnerable. Well, we've never been vulnerable.

Leo: Never ever because it's IIS.

Steve: Never ever, exactly.

Leo: Broken pipe. I don't know what that means.

Steve: So he's saying "broken pipe." He's not saying what that means. People are interpreting it incorrectly. Now, to his credit, and I was going to mention him, and we can right now, he was up first. He did it fast. Unfortunately, his site just can't handle the traffic. You can imagine the frenzy of everyone trying to go there. What he has, though, if you click on the GitHub link in the upper right corner...

Leo: It's working, yeah, yeah.

Steve: Well, he's got a command line version. So I've already had requests from people saying how can we test our Intranet? An external service doesn't help. So this would allow you to use his tool on internal Intranet servers to check for this problem. Or also to do your own tests on Internet servers from your own command window. So it's useful.

Leo: It's written in Go, which is a Google language.

Steve: Yeah. There was some - there's also some Python involved. And I guess he was saying that his Python - there was a Python script which he believed at one point was collapsing under the load. So it may be good. It may take him a while to get it solidified. I imagine other people will put things online immediately, too. I don't need to. I couldn't, anyway, because I'm way overcommitted, as we know. But I'm super happy that Ivan has got SSL Labs doing it because I trust them to do it right. His initial concern when we began talking was he didn't want to trip intrusion detection systems that might have been looking for misbehavior. He's very sensitive to his test being industrial grade. And so when it comes from him, I absolutely trust that he did it the right way.

Now, what's the actual percentage? The best thing, the best information I've seen is from Netcraft, that have been profiling continually, for many years, the Internet's servers. There's a link in my show notes, but I also have a huge pie chart in the show notes. And of the SSL-supporting sites, 82.5% do not offer the extension. They aren't and never were vulnerable. So even though two-thirds are potentially vulnerable because they are Apache and nginx servers, what we know from Netcraft is, by their estimate, only 17.5% of SSL connection-accepting servers are vulnerable. And I urge anybody who's interested to get the show notes, scroll down to this big pie chart, click the Netcraft link, or maybe you can find it from the home page of Netcraft. It's certainly - it's recent, so it ought to

be at the top of their stack of information.

Also in the show notes I've got two different links to source code diff displays for anybody who wants to look at it, very much like the Goto Fail problem, exactly what it was that went wrong and how, and the 1.0.1f code right next to the 1.0.1g code which has it fixed yesterday. There's also a very nice blog analyzing the bug. Our friend @e_StrategyPro, who is a Twitter follower, sent me this, and so thank you for that. That's at Existentialize.com, the blog.existentialize.com, and I have a link in the show notes, where he basically takes us through the program, this phase of the program logic, loading a pointer and then dereferencing the specific information from the packet and so forth. So anyone who wants to get more geeky into this, there's information there.

The LastPass guys responded immediately with a - by fixing the problem last night, they updated OpenSSL, they restarted their servers to load it, they revoked their old certificate, got themselves a new one immediately, and then put up a blog posting explaining that, fortunately, due to the fact that they are TNO, Trust No One, all they're doing is storing blobs for people in order to give us iCloud connectivity. We keep the keys. And this of course is the reason why I selected and personally use and trust LastPass, is even in the face of this kind of significant potential dilemma, we were safe as a consequence of the architecture which didn't require that we trust SSL connections. So that's really good. Even if someone were to spoof, they would just get a blob of data that they can't do anything more with than the LastPass guys can.

Leo: Go ahead. Because I know what you're going to talk about next. So go ahead.

Steve: Yeah. So that's - so where are we left? The Internet servers have a dilemma. What I think we really need is every site which depends upon SSL/TLS security needs to tell its users, its customers, what this means to them. I mean, even non-techie sites. Bank of America. Twitter does have a blog post up contradicting what I saw earlier, saying that this was never a problem for them. It may have been that Netcraft noted that they, well, if Netcraft noted - oh, maybe they're using IIS. So Netcraft saw that they were using the TLS extension, but in their case they weren't vulnerable to it. I don't know. But what we really need now, because as users of sites, our trust, through no fault of theirs, is called into question. And so we really need statements from those sites saying this is what we know, what little we're able to know, but it's more than we as users know. We need some sort of response from individual Internet hosts.

Leo: But what do we do? I mean, if Twitter says, yeah, should we change our password? Is that what you need to do? What do you do?

Steve: Well, okay. So the biggest problem is with the fact that there wasn't private notice. Remember that, for example, when Kaminsky, when Dan found the problem with DNS servers, he was able to work in secret with the DNS server community to fix the problem with nonrandom - I can't remember what it was. There was a nonrandom something in the query, 16 bits. I have my whole spoofability page that's all about that. But so he was able to fix the problem and get everything deployed and largely remediated before he went public with it. Here the problem is everybody found out about this at once. And, I mean, the nature of the problem, the fact that GitHub has source code checking for and demonstrating it means that bad guys are almost certainly in a mad scramble to build a tool and immediately start sucking down 64K blobs of data, even to look at later, but to get it now before websites update themselves to a non-vulnerable

version of OpenSSL.

So the biggest problem is this, like, who's going to get there first? Are sites going to get themselves fixed before the bad guys are able to reach in and get their credentials? So we as users, with the exception of the problem with Perfect Forward Secrecy, which was probably only useful to the NSA because, if the NSA was able to get certificates for websites, then they could suddenly decrypt all of the traffic, encrypt that traffic that they presumably have been archiving over all these years. And we're assuming they don't already know about this. Who knows? But with the exception of that, it is only, for us, connections that we make from this point until the website has secured itself that are at risk, and only to the degree that we could be misled to a fraudulent site, either through a man-in-the-middle attack impersonating a site or a DNS redirect of some sort that has us going to the wrong IP, but our browser doesn't know. The other thing we need to do is to turn revocation on.

The last link that I have on the show notes is to a blog posting - and Leo's got it on the screen now - where it says in Chrome you have to go into settings, and under HTTPS/SSL there's a Manage Certificates button. Right below that it says, "Check for server certificate revocation." That is normally off. And I haven't had a chance to check Firefox, but I immediately will. We should all turn that on because even certificates that have escaped from the control of their owners well before expiration - remember that all certificates ultimately expire.

So stolen certificates will expire. And this is another reason why I'm no longer complaining about certificate expiration, even though it's expensive for me. It's clear that having a two-year or three-year horizon on the inherent life of this cryptographic credential is important because that means that revocation only needs to override what would otherwise be the certificate's acceptance until the certificate would have already otherwise expired, in which case the expiration will prevent the browser from accepting it. So expiration works. Revocation can - the reason it's turned off by default is that it can slow things down a bit. You may need to be - and we did a whole podcast on revocation, so anyone who wants the details can go back and find it [SN-295].

But either you go to the certificate authority and get its information, or there's OCSP, which is the online certificate revocation-handling protocol that allows a browser to dynamically query for that. And we've talked about, though, the problem, for example, is some browsers, when they don't get an affirmative reply, they assume it's okay because they didn't hear it wasn't, even though it might not be. So really for the first time I'm not sure - we've seen individual small instances of certificates which have had to be revoked. Here we're in a situation where potentially every website that might have been in danger, I mean, like LastPass we know already did the right thing. All the other websites, 17.5%, apparently, that had this extension enabled, if they're going to be responsible, they will revoke the certificate they've been using and issue a new one. Well, that's only useful to us if we are smart enough not to trust the revoked certificate. That's the one that we're now vulnerable to.

So the real takeaway from the podcast is we need to make sure revocation is working. And it's funny, I was just thinking, okay, where can we - if anyone knows of a site that tests for revocation. I was wishing that I had one, but I don't. That would be very useful because it'd be neat to be able to go to a site where there is a revoked certificate and be able to verify with our browser that, okay, good, revocation is working here because, if certificates got stolen, until they naturally expire, we're in danger if we're not checking for revocation. And we're not, normally.

Leo: Unfortunately, mobile browsers apparently don't check for revocation. And I don't see any way to turn that on. And since most browsing now is - more than half is mobile, it's going to be a pretty widespread vulnerability.

Steve: Yeah. And moving forward, as I mentioned, the danger is that we would be using a site that has not fixed itself, and our credentials could be vulnerable if the certificate had been exfiltrated. But mostly I think what we need to do now is get ourselves tuned up for a major revocation fest.

Leo: Mmmmm.

Steve: Yeah.

Leo: Apparently in a Mac Keychain on the desktop there's something called "Online Certificate Status Protocol."

Steve: That's the OCSP, yes.

Leo: And that Safari pays attention to. So that would be sufficient; right?

Steve: Yes.

Leo: Because all the revoked certificates would be in that database.

Steve: Yes. I expect that were going to see a flurry of revocation from responsible sites, and we just need to make sure that our browsers pay attention to them. And I love the idea of honeypots being established because, again, because this thing leaves no trace, nobody knows currently if, you know, the degree, if any certificates have or were stolen during this two-year window. So, boy.

Leo: This is the setting, if you go to the Mac Keychain, for OCSP. It's current - the default is "best attempt."

Steve: Ah, yes. And so that means - that's what I was referring to. It tries. And if it doesn't get an answer, then it says, oh, okay, we'll just go with it. Yes.

Leo: So we should say...

Steve: Require.

Leo: Require, well, there's only "Require if certificate indicates." We can't select "Require for all certificates." So I don't know.

Steve: Well, I will clearly be doing some more research on this.

Leo: This is not in Safari. This is in the Macintosh Keychain access program.

Steve: I wonder, then, if - the option is there. I wonder why it's grayed out?

Leo: Yeah. Because what you would like is require for all certificates, right, that it check with the OCSP.

Steve: Yeah. You know, "Require if certificate indicates" is probably okay because the certificate itself, and probably all - see, remember there are different types of certificates. There are not just SSL authentication certs. There are many other flavors. For example, Authenticode for digital signing of drivers to allow operating systems to trust them and so forth. Certificates specify, in the certificate, where you go for the revocation information of the cert. So I would imagine any certs we care about will be specifying where to check for their own revocation.

Leo: That would make sense.

Steve: And, yes, and remember, the bad guys can't change that. So the bad guys got a certificate that tells you how to check for its own revocation. It's been digitally signed, and that's why we trust it. And encapsulated in that signature is the OCSP or other revocation information. Sometimes there is a URL that tells your browser, go here to check for this certificate's revocation. So that certificate will come to our browser if we've told it to "Require if certificate indicates." What that means is the certificate has said here's where you go to check for revocation. So what we need to find are a source of revoked certificates so we can verify.

Leo: And that's what OCSP is. Now, HighFive [ph] said, if you hold down the option key - I guess they just don't want you to kind of turn it on unless you really want to do it.

Steve: Oh, my goodness.

Leo: So it's grayed out unless you hold down the option key. Then you can say "Require for all certificates." That's what you want; right?

Steve: Yeah, I would. I would run that way. I mean, we need experience with this. This has never been such an issue for us as it is today. So we'll all be developing some experience in, you know, is it like running with NoScript, which annoys you, Leo, because

so many things break. Maybe this will, in which case you'll want to back off. But I would say for, like, immediately, turn on "Requiring for all certificates" and see how that works.

Leo: So I've turned that on in Keychain Access. That will cover Safari. I've turned it on in Chrome by going to the Chrome Advanced settings, scrolling down to the HTTPS/SSL section and checking that box. And then Firefox has something similar; yes?

Steve: And remember that what this means, then, is that no site would revoke the certificate they're using. They're revoking the one they were using. So if you ever get a revoked certificate from a site that you're legitimately going to, that's, I mean, there's no reason that should ever happen. That's happening because somebody redirected you and is trying to spoof you.

Leo: Right. And there is a Windows utility, and there might even be on the Mac, that will do this. But that's a pain. You don't want to have to go to every site you're going to go to and check before you go to the site. You really want it in the browser.

Steve: Well, yes. So the solution is we need a revoked certificate somewhere. I'm sure somebody will come up with one, or maybe there is a test site.

Leo: To test it, yeah.

Steve: Yeah, to test. We need to just verify that, when the browser we're using, whatever browser we've chosen, encounters a revoked certificate, it tells us. And once we know that, then we're okay. And then, I mean, we're as okay as we can be because now we're still relying on the proper behavior of the web servers. Anyway, this is a perfect example of the worst possible kind of vulnerability because it is in a public-facing, hugely used protocol. It's not a malformed image, where you have to get the right version of some, well, like, for example, Office and a malformed RTF. I mean, it's not content. It's a public, hugely used, openly accessible, Internet protocol. And it's the actual, you know, the protocol behind the port allows essentially the OS to have 64K of its innards exfiltrated. I mean, that just - it'll keep security guys up at night.

Leo: Wow. Heartbleed. And now we know how it works, what to do about it. Thank you, Steve Gibson.

Steve: Absolutely. Next week we'll do Q&A, and then we'll talk about harvesting entropy in two weeks, which I know everyone's going to find very fascinating.

Leo: If you want a question for Steve, this would be a good time to go to GRC.com/feedback and leave that question. We'll be taking 10 questions next week. You can also, while you're there, pick up a copy of SpinRite. Wouldn't hurt. World's best hard drive maintenance and recovery utility. You can also find lots of freebies at Steve's site, Perfect Paper Passwords and all of that. That's GRC.com. 16Kb versions

of this show are there, audio only. He also has text transcriptions. We have higher quality audio and video at TWiT.tv/sn. Of course you can always subscribe wherever you'll find the better podcasts. We're right there. Just look for Security Now!. We do this show Tuesdays at 1:00 p.m. Pacific now, that's our new time, 4:00 p.m. Eastern time. Is it working out for you, the new time?

Steve: Yeah. I like it a lot, Leo, really, really do.

Leo: Gives you time in the morning to prepare. You don't have to rush around.

Steve: Yup, yup.

Leo: All right. Steve, we'll see you next week for a Q&A, barring any breaking security news. I'm going to - I just got my copy of Windows XP, and I'm going to go install it on all my systems because you said it's safe.

Steve: It'll be fresh, Leo, a nice, fresh install.

Leo: Yeah. We're going to have a little fun.

Steve: Whatever you do, don't put that one unpatched on the Internet or it'll get taken over immediately by Code Red or Blaster or Nimda or any of these other things. Because, remember, it had a rough start.

Leo: Oh, yeah.

Steve: I'll never forget. Remember Ballmer strutting around: "Windows XP is the most secure operating system we've ever produced."

Leo: Oh, lord.

Steve: And my comment at the time was wait a minute, you can't state that ahead of time. History. The only way you can do it is retrospectively. And, boy, it had a very rough time.

Leo: Ironically, you probably could say that today because it's been patched so much, and all the obvious holes have been fixed.

Steve: It's 13 years, and it's got its firewall on by default.

Leo: Right.

Steve: And that's my position. We'll see whether I'm right.

Leo: We'll find out tomorrow, yeah.

Steve: But I'm sure that next week we'll have follow-up for this, everything that has happened in the week intervening on this catastrophe. And keep an eye on my Twitter feed because I will let people know as more news evolves. This is, as they say, a rapidly unfolding story.

Leo: I hear people tweeting you as we speak.

Steve: Thanks, Leo.

Leo: Thank you, Steve. We'll talk again next week on Security Now!.