Listener Feedback #78

Description: Steve and Leo discuss the week's major security events and discuss questions and comments from listeners of previous episodes. They tie up loose ends, explore a wide range of topics that are too small to fill their own episode, clarify any confusion from previous installments, and present real world 'application notes' for any of the security technologies and issues we have previously discussed.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-220.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-220-lq.mp3

Leo Laporte: This is Security Now! with Steve Gibson, Episode 220 for October 29, 2009: Listener Feedback #78.

It's time for Security Now!, the show that covers all things secure - online security, the browser, privacy and more. With us Mr. GRC, the Gibson Research Corporation head honcho, Steve Gibson. Hey, Steve.

Steve Gibson: Hi, Leo. Great to be with you again, as always.

Leo: Good to see you.

Steve: Episode #220.

Leo: Wow, that's half of 440.

Steve: Well, we're halfway to 440.

Leo: Halfway to middle C. So, or is that A? No, that's A, isn't it. 440's A.

Steve: A is 440. A below C, yup.

Leo: Halfway to A below C. 220's probably just A, a couple octaves down or something.

Steve: It would be one octave down, yeah. Since an octave is a doubling of frequency, or halving.

Leo: Are you a musician? You obviously know this stuff. I didn't know that.

Steve: Yeah. Yeah, a lot of software people are.

Leo: I know.

Steve: I was fascinated by synthesis in the early days before it was really mainstream. You know, Robert Moog and all that stuff.

Leo: You made a synthesizer in college.

Steve: I did. In fact, it was one of the things that caused some angst with me because, as a student of the UC system, any intellectual property created by the students or the faculty is the property of the Regents of the State of California. And that just really seemed wrong to me. I was inventing stuff all the time. It's like, wait a minute, you know, how can I be here...

Leo: I'm paying them.

Steve: [Laughing] Exactly.

Leo: They're not paying me.

Steve: Exactly. I came up with a way of storing analog waveforms in a digital shift register. So it was probably one of, as far as I know was the first sampling synthesizer. And I showed it to one of the professors of the electronic music lab at Berkeley whose jaw just dropped open. Because I just - I went "ah" into a microphone, and it played back "ah ah ah ah ah" [ascending scale] on the keyboard. And he said, "Vowel sounds."

Leo: Now, that's cool.

Steve: And it turns out that with this whole wall of - he had all these Moog modules. And he said, "All of this, and we can't make vowel sounds." I was like, "Oh, okay." He said, "Can it do E?" I said, "Yes, it can do E. It can do anything you want."

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

Steve: Yeah.

Leo: Oh, that's so funny. That's cool. Did anything ever come of it? Did you ever...

Steve: No, no. I got all busy doing other things and got distracted and never got back to it. And of course then the world discovered sampling synthesizers, and that's all we have these days.

Leo: Right. Wow, really cool.

Steve: So, yeah, that was back in '73, yeah.

Leo: Very cool little bit of history there. And another surprise about Steve Gibson. You just, you know? You're full of surprises, Steve.

Steve: Well, yes, music - music and programmers, of course, goes hand in hand.

Leo: Oh, yeah.

Steve: There's a lot of programmers who are major musicians. I'm no big-time musician. I took piano. Mom made me. And I'm glad for it now. I have a little more appreciation than I would have otherwise. But never pursued it.

Leo: It's just a left-brain dominance or something.

Steve: Yeah. There you go. Or right brain.

Leo: Right, left, right, I can't remember which is what.

Steve: Because I am left-handed, so I'm right-brained.

Leo: Okay. Logic is left, and emotion is right.

Steve: Correct.

Leo: Okay. Correct. Right brain is - I'm looking now - random, intuitive, holistic, synthesizing, subjective, looks at wholes. Left brain is logical, sequential, rational, analytical, objective, looks at parts. Hmm. I don't know what that means. It's not exactly astrology, but maybe close.

Steve: As long as you - and what you want is you want the two halves connected, so they can talk to each other.

Leo: That's very important.

Steve: Yes.

Leo: Bad when they get disconnected. Today a Q&A, our 72nd in a - or, I'm sorry, 78 in a continuing series.

Steve: Yup.

Leo: That's exciting. Before we get to that we're going to get some security news. We'll also get some addenda and errata, should there be any.

Steve: Yes, we got 'em, we've got 'em.

Leo: Addenda and errata, we got 'em.

Steve: So literally, as I was preparing the notes and gathering things for recording this with you, like an hour ago, Leo, a pop-up came on my screen saying that there was a new version of Firefox. I hadn't run across it in the news since it was literally, well, it was two days ago when this show airs, but it was yesterday for me as we're recording this the day before. So I just wanted to let people know that Firefox moved, the v3 train moved from 3.0.14 to 15. So it's now 3.0.15.

Leo: Oh, Steve, Steve, Steve. We're up to 3.5 now. You're still using an old version of Firefox.

Steve: I do. I have 3.0.15 on one machine. But I am at 3.5 on several others.

Leo: 3.5.4 is also the new update, just came out.

Steve: Correct, correct. And what's interesting is that you can really see, if you look at the changes between these two versions, which I did, they're almost the same. So this demonstrates a huge block of shared code between the 3.0.15 and the 3.5.4.

Leo: Do they say what's getting fixed in 3.5.4 at all?

Steve: Yeah. There was a - both of these versions, the older and the new, had a crash with evidence of memory corruption, which makes people nervous because that's the kind of thing which, okay, it's a crash today, but it's an exploit tomorrow.

Leo: Right.

Steve: So they fixed that. There were some what they called "memory safety bugs," which they weren't more clear about, but in their media libraries, which were updated to fix those. There was a heap buffer overflow in string-to-number conversion. And remember that we talked about a library a couple weeks ago that was heavily used that had that problem in it. So they're probably catching up with that also. There was a privilege escalation bug they fixed; another heap buffer overflow in the GIF color map parser. And form history was vulnerable to stealing. And so they closed that hole. And all of those were common to both versions of Firefox, showing that there's a lot of code that they have in common at this point. And the v3.0 is being phased out in favor of 3.5.4, which has faster JavaScript processing and other stuff. So just wanted to let our listeners know that their Firefox needs to be updated, if it hasn't told you itself already.

A huge blurb in the news in the intervening week since we talked last, and a bunch of our - I saw many of our listeners wrote in to make sure I had run across this. And that is, a consultant working for one of his clients stumbled on a big problem with Time Warner's, their standard WiFi cable modem router. It's the model SMC8014. So anybody listening who is a Time Warner subscriber who has their default SMC8014 WiFi router, or knows somebody who does, needs to sort of sit up and pay attention to this. It turns out that the web interface to the router, that is, the interface that the router users would use, was being hidden, a large chunk of it was being hidden only by the JavaScript which the router was assuming that the web browser was running.

So this consultant was having some sort of problems that for whatever reason caused him to disable JavaScript on the browser. Suddenly, a whole 'nother bunch of UI showed up. All the admin side of the router which JavaScript was being used to obscure was then visible. And by poking around a little bit he found that there was a backup file of the original router settings, and that the router was still using its default admin username and password, and that this router was exposing port 80, the standard web interface port, on the WAN side.

So the good news is, this is the kind of thing that just running ShieldsUP! at GRC would show you, unless Time Warner was blocking port 80. But I know they're not because other people, in hearing about the story, started doing port scans of the Time Warner IP space, and all of their customers have port 80 exposed because all of their customers with this SMC8014 WiFi cable modem router are publishing their web interface to the WAN side. And as you and I have said to people, Leo, over and over, I mean, that's one of the first things you want to do when you're configuring a home router is make sure that by default you don't have the WAN admin port enabled and opened.

And again, it's easy to check. You can use ShieldsUP! just, you know, quickly to make sure that you don't have any ports exposed. And of course the vulnerability is that bad guys are able, it's been conjectured, maybe even able to change the firmware on the router from the WAN side. Time Warner is scrambling madly right now and working with

SMC to come up with an updated firmware that will fix all this. But this is just a big faux pas. That's not good to find out.

Leo: Yeah. We talk about that setting. Always turn off WAN administration.

Steve: Yeah, unless...

Leo: I wonder if that's for their techs, so that the techs could get in there?

Steve: You know, I just have to think this is just sloppiness. I mean, first of all, it's flaky to depend upon JavaScript, in the pages that the router is serving to the client, to depend up on JavaScript not to see the admin interface. That's just bad design. And so if you've got that going on, lord knows what else is happening. I mean, that just sort of says, okay, we're not really up to speed here on the design of our router from a usability and security standpoint. That's crazy.

And then I did want to note also that there's activity over on the FCC and the Net Neutrality front. The FCC has published a 107-page, what they call their "Notice of Proposed Rule Making," the NPRM, which is basically a call for comments from the industry, from organizations, from commercial providers, even from us individual users. They have a 60-day comment period which closes January 14th. And the good news is that, if this goes as we hope it's going to, they're talking about officially putting into law the kinds of Net Neutrality enforcement that people who really understand the 'Net are hoping we're going to get. Which basically will, as we've talked about before on the whole Net Neutrality idea, is prevent providers from being anticompetitive, essentially. For example, for, like, throttling Skype connections if Skype competes with their own proprietary VoIP offering, doing that sort of stuff. The idea being that they'll not be able to perform any sort of unfair restraint of trade, filtering of content without there being some clear justification and specifically without clearly letting their users know exactly what they're doing. So that's all good news.

Leo: I like what they call this. Because "Net Neutrality" really confuses people.

Steve: It's a bad term; you're right.

Leo: Yeah. So they call this, and I think this is good, "Preserving the Open Internet." That's the name of the NPRM. I think that that's a better description of what we're talking about here is keeping it open, preventing discrimination.

Steve: Yeah.

Leo: Good. Well, are we allowed to comment?

Steve: Yeah, yeah. I mean...

Leo: Anybody is.

Steve: Yeah, anybody is. I don't know how much weight individual voices have, as opposed to organizations like the Electronic - the EFF, for example.

Leo: I'm sure they'll say something, yeah.

Steve: Yes, absolutely. And, you know...

Leo: As will John McCain, so, you know.

Steve: And Vint Cerf...

Leo: Vint Cerf.

Steve: ...is very vocal on this. And unfortunately, I mean, then on the flipside AT&T and the 3G consortium have - they're already making lots of loud noises, saying, oh, this is going to impede consumer choice and prevent us from getting the leverage out of the bandwidth, the airspace that we've been promised, and blah blah blah. It's like, okay, well, you know, I'm glad somebody else has to have the headache of sorting all this stuff out.

Leo: Well, yeah, but I'm glad to hear also everybody's input. Maybe there's a good reason why we should tone the regulations down or not have regulations. I mean, I understand people's distrust of government regulation, especially on something as fast-moving as the Internet.

Steve: Yeah, I agree. Hearing from everyone is just a good thing. And the other thing that I thought was really interesting, and I'm sure you've picked up on this, I'm beginning to - we're beginning to see the notion of Internet access being referred to as a basic human right.

Leo: Yes. Right on.

Steve: Which I think is really interesting. There's now a big push in the United Nations to formally declare Internet access as a human right. And apparently in Estonia, France, Finland, and Greece it is already recognized formally as a - Internet access as a human right. Which is like, wow, look how far we've come.

Leo: Well, do you agree? I think it - I agree. I mean, I think nowadays without the Internet you're kind of left out.

Steve: I agree. You're disadvantaged.

Leo: Yeah. The conversation goes on without you. And some very important conversations, like political conversations and policy and so forth.

Steve: Yeah. To me it feels like a big step. That's like a big step. And then of course if you declare it that, then I wonder what that means politically and economically. I mean, does it mean that everyone has a - if they have a right to Internet access, well, from a technology standpoint that's still not free.

Leo: Well, and how do you do it; yeah.

Steve: Yeah. Yeah. And then one little blurb from last week that I forgot to mention. We were talking about eBook readers and Kindles. I thought it was interesting, just wanted to bring up to our listeners' attention that Amazon has stated that Kindle buyers purchase 3.1 times as many books while they own a Kindle than they historically did before. And I have to say, I'm part of that statistic. It happens to me. There's a sort of a sense of, oh, gee, I want to read that. There's a little bit of that immediate gratification. Or it's like you're searching around, and you see books you wouldn't have stumbled on before. It's like, oh, interesting. I think I'm going to grab that. So I know that I've sampled a lot of books. So I thought it was interesting. But Kindle buyers are buying 3.1 times as many books as they did before. I wonder how many of those they're reading, or if they end up with unread books piling up?

Leo: I have unread books in my Kindle, I have to say.

Steve: Yeah, I do, too, yup.

Leo: Because it is so easy. And I'm willing to bet that the same thing, if you did a study of iTunes music users, same thing. Ease of access changes everything, doesn't it.

Steve: Yes. Yes.

Leo: If you can - and let's face it: Amazon, with the 1-Click thing, knows that pretty darn well. I mean, they make it so easy to buy stuff. Just click, and you own it.

Steve: And the next screen is "Thank you for purchasing."

Leo: Just instant.

Steve: Wasn't that nice.

Leo: Who cares how much it costs?

Steve: Yeah, yeah. And then I had a fun little blast from the past. A listener, and a past viewer of ours, Leo, from The Screensavers, Kelly Stowell, said, "While I knew of SpinRite since The Screensaver days, I always told myself I didn't need it, since I swapped out hard disk drives at least every year. Alas, my 'no worries, I can always afford new hardware' days are over. I'm looking to get a new SSD drive" - which we talked about last week, I think, or the week before, the Solid State Disk drives, which are coming down in price, still much lower capacity than their spinning-platter cousins, and about 10 times more expensive, maybe? And interestingly, not that much faster, Leo. I just swapped out a 75GB on my little ThinkPad, my X61s, for that 64GB SSD. And it's like, yeah, I mean, I'm doing it for reliability. But I don't see a huge difference in speed. I see no...

Leo: I see massive difference in speed.

Steve: Do you really. I wonder if it's a Mac versus a PC thing.

Leo: It might be a Windows issue. Make sure that the computer you're using this on - I had this issue because I put it in some Netbooks. It has to support the SATA2 spec, or you won't get the speed benefit. So in other words, if you don't have the throughput, and I bet you you don't on that XS...

Steve: On the little X61s?

Leo: Yeah, because that's a few years old.

Steve: Yeah.

Leo: So you're not seeing - in other words, the drive's faster than your pipe.

Steve: Right, than the interface is. Yup, that makes sense.

Leo: Because you do see benefit, of course, because the access time is virtually zero because it's random access.

Steve: Yup.

Leo: And the read speeds are very, very fast. And I'll tell you, on the Mac, I boot from it, and the boot time has come down to almost nothing after the power-on self-test.

Steve: Oh, nice.

Leo: Once it actually starts, it hits the drive, it's [sound effect], we're here.

Steve: Ah, cool.

Leo: Now, I'll tell you, I just ordered a new Dell laptop for running Windows 7 that only has solid-state drives. And I'll let you know if I see a similar thing. It may be that Windows, because of the nature of its boot process, maybe there's a lot of thinking involved and stuff, might not benefit. But, boy, the Mac does. You don't see, I mean, applications launch instantly.

Steve: Wow.

Leo: And it's - you get kind of used to it, I have to tell you. I have to tell you.

Steve: Well, so he says he's "looking to get a new SSD drive, but not for another five months. And my current drive has to hold me over until then." And he said, "Unfortunately, my not-very-old Raptor drive began to fail, and I hoped SpinRite might help. And considering that it was a Raptor," which I guess he must mean was expensive, "it was more affordable for me to get SpinRite. Well, SpinRite worked very well and got me back up and running within two hours. Excellent product, Steve. Regards, Kelly Stowell, Windsor, Ontario."

Leo: Isn't that nice.

Steve: So thank you for the feedback, Kelly. Appreciate it.

Leo: Another happy SpinRite customer.

Steve: Once again.

Leo: Once again. Well, it's not surprising. We use SpinRite on everything here before we use those drives, before we put them into use. Because it gives me...

Steve: Do it preemptively.

Leo: Yeah, it just gives me that sense of confidence. We really hit our drives hard.

Steve: Yeah. Oh, you're using them like crazy.

Leo: Oh, we go through a lot of them. But also we record video direct to them, I mean, it's really - they're getting exercised. So it's important to us.

All right, Steverino. Are you ready, my friend, for questions? Endless questions for Steve. But we've picked 10.

Steve: And #10 is the Security Screw-up of the Week.

Leo: I love those. Oh, I love those. Starting, though, with Marv Schwartz at Case Western Reserve - very good school, very good tech school - commenting on the Mozilla/Firefox plugincheck that we talked about last week:

Steve, by now you already know, I would guess, that in order to get www.mozilla.com/plugincheck to work with NoScript - that's that plug-in that you use, that you recommend to protect yourself against JavaScript exploits - you have to allow both www.mozilla.com and - and here's the little catch - www.mozilla.org.

Steve: Yes.

Leo: That's where the plugincheck comes from.

Steve: Yes, exactly. And I've mentioned to you when we talked about this, and again I recommend this for Firefox users, it worked for you, it wasn't working for me.

Leo: Aha.

Steve: And I had enabled Mozilla.com. But then if I'd just looked again at the little NoScript icon, I would have seen that it said, oh, there's something else that I'm blocking. And then you click on it, and I did when I saw his note. It's like, oh.

Leo: Duh.

Steve: Doh. And then I enabled Mozilla.org; and, bang, it all worked perfectly. So if anybody else got caught out by that, whom I've recommended NoScript to, who are NoScript users - and I know many of our listeners are because they really do, they're as concerned as I am about the issue of scripting - that that's the little catch. They probably were a little more on the ball than I was and noticed that Mozilla.org also had to be enabled. They are now both for me, and the plugincheck works great. And it turns out that I did have one out of 12 that was back versioned. I had an older version of a QuickTime plug-in for Firefox that was no longer current. And so it was like, oh, good, how would I have known that otherwise? So, as we know, when they go to the next major version of Firefox, which will be 3.6, it will incorporate this automatically. It'll show you if plug-ins on the pages you're visiting are back versioned. It'll let you know that. But again, Mozilla.com/plugincheck is a great service from the Mozilla guys. But you do need both Mozilla.com and Mozilla.org enabled for scripting.

Leo: Is there a pop-up that tells you, or something, that you're getting content from a different page?

Steve: Well, actually on NoScript the little icon notifier down in sort of the equivalent of the tray down there in the lower right, if it's blocking the main page you get like the big red slash through the icon. And then, if you enable the main page but subsidiary things are being blocked, it's a much smaller - in my case it was a little too small, but when I knew to look for it, it's like, oh, of course. It's just sort of a smaller thing saying the page you're visiting we're allowing scripting on, but we've blocked scripting from somewhere else. And in this case it was scripting from Mozilla.org. And it's, again, trivial to enable it. And I made them both sticky because I trust Mozilla.com and Mozilla.org. And this way, of course, with NoScript, if I'm rambling around the 'Net somewhere, I don't have scripting enabled.

Oh, speaking of which, I want to confirm that we do have John Graham-Cumming, who had to back out last week from our having him on our planned episode 221. He will be on 223. So he is our guest next week to tell us in depth about JavaScript security.

Leo: Oh, excellent. That's fantastic. Yeah, I can't wait.

Steve: Yeah, it's going to be great.

Leo: Question 2, Paul in London, Ontario, Canada wonders about making online banking safer. He says: Hello, Steve and Leo. Long-time listener. Love the show. You both are doing a great service, and I appreciate your podcast every week. Thank you very much, Paul. My question is my bank is offering a program called Rapport by Trusteer to help protect my online banking transactions. I was just wondering if you have any information you could share about the program, and if it's needed when I do my online banking. It raises the question of why is my bank offering this? Don't they think their security measures are enough? The bank I use is President's Choice Financial in Canada. Thanks in advance if you use my question on the show. You guys are great.

And Steve, could you make your own OS? Call it SOS - Steve's OS - and it can save all of us from the other choices. I'd use it. An OS built from the ground up with security in mind? P.S.: I have a copy of SpinRite. I got one problem, though, Paul. It'd probably look like DOS, I've just got to warn you. Certainly be a command line OS. I have a copy of SpinRite, and it has saved my bacon a number of times. Great product, Steve.

Steve: Yeah. The problem with Steve's Operating System is that it would - we'd all be old and grayer than we already are.

Leo: Well, you're going to write one, aren't you, for the PDP-10? Or 11?

Steve: I'm going to write one. But no one will care. I mean, you know, except five other people who still have old PDP-8s alive and running. But...

Leo: Oh, that'll be fun.

Steve: It really will be fun.

Leo: That's a good - that's your retirement hobby.

Steve: It's my retirement hobby, exactly, exactly. But, so, yes, unfortunately not mainstream on the OS side. It's just too big a project, really, for one person. And to do the kind of job that needs to be done, it would take forever. But relative to Paul's question, this Rapport by Trusteer is something I've run across a few times. And it's interesting. It's an alternative to what we've been talking about. We've been talking about the fundamental problems of the browser and server security. So this Trusteer is a company that's a third-party offering that hardens browsers on behalf of their clients. In this case their client is the bank. So the bank offers this Rapport service. And essentially it is a plug-in, a toolbar that you add to your browser. And what it does is it basically does everything they can think of for hardening your browser.

For example, it's very much like sort of now we have in the most recent browsers we've got private browsing where history of the things we do are not left behind on the machine. Nothing is written to the hard drive or into the file system, but it's kept in RAM. They actually - they have some DNS hardening technology so that you're not prone to DNS spoofing. They don't go into great technical detail about what they've done. But they make it very clear that they understand that the openness of the APIs in our contemporary browsers, which is what allows toolbars to know where you're visiting and what you're typing, you know, those kinds of APIs are being leveraged by hackers in order to gain access to what you're doing.

So this is a very good thing. This is something, when Paul asks why is my bank offering this, don't they think what they're doing is enough, I would argue that the bank recognizes they don't have the kind of control over the other end of their connection. They can have a super-secure server and have their end all bolted down really well. But if the user's got malware in their computer that's doing keystroke logging and things, basically the bank is having to rely upon the integrity of the browser which they're using to interface to them. So I like this idea. Instead of, for example, the bank developing their own wacky individual application to talk to them, they're saying, okay, we're going to use a third party who's got all the technology, bringing this technology to the table of hardening the browser. So I think it's a great thing. And they've got a whole bunch of banks that are lining up behind them and using this technology. So...

Leo: This is good. I didn't realize when you mentioned it. This sounds like something really good.

Steve: Yes. I mean, this is a beautiful reaction to the fundamental problem that we've been talking about for the last couple weeks of the whole browser model just being prone to abuse. And so these guys are coming along and saying, okay, we're going to - they use words like "vault" and so forth to say we're not letting your data escape through the browser APIs, where the openness of the API is normally something that allows you to leverage the power of the browser, because in this case you don't want openness. You want this thing to be closed and bolted down while you do banking. And so this is an

add-on for browsers. I think it sounds great.

Leo: Yeah, I'm kind of a fan of sticking to your knitting. And if somebody's really good at security, they become the people who do it; right?

Steve: [Laughing] Sticking to your knitting.

Leo: Sticking to your knitting. The bank should do what it does best. Every bank cobbling a solution together is not a good idea. I think you're exactly right.

Steve: Exactly.

Leo: And then we only have to vet one solution and feel secure with that.

Steve: Yup.

Leo: That seems like a good way to go.

Steve: In fact, we'll be talking to sticking to your knitting here in question #4, as well.

Leo: Good, okay.

Steve: I love that.

Leo: Before we get there, Abhi Beckert in Cairns, Australia has a Mac OS "10" tip: Hi, Steve and Leo. Have you heard of the ClickToFlash plug-in for Safari on the Mac? I haven't. It's a free open source plug-in which disables Flash by default, replacing all Flash objects in the page with a simple box. Then you click to load the actual Flash object. There's something similar on Firefox on the Windows and Mac side, as well, I believe. I'll find out for you. Apple has reported that Flash causes more crashes under Mac OS X than every other Mac application combined. Now, we should mention that Apple doesn't like Flash and has been trying to kill Flash for some time. It honestly doesn't surprise me that Flash has so many security holes.

ClickToFlash reduces Flash use only to those times when I want it - a video, a photo gallery - and eliminates Flash banners and Flash cookies altogether. Oh, I'm going to have to install this. I never allow those Flash objects to run. It's a great security tool. I use it on both my desktop and laptop. As a side benefit, my laptop runs noticeably cooler and with better battery life. I think there is a similar plug-in for Firefox.

Steve: Yeah, so I just wanted to mention that. I wanted to notify you and the Mac users who are using Safari who didn't know. This is the kind of thing which I just think is a good idea. It's going to lower your bandwidth. You're not downloading Flash objects.

Flash, I mean, Flash is one of our constant problems that we're reporting from a security standpoint. So if you don't mind your page having, like, big dead spots all over it where normally all kinds of Flash animation is running, I know it's a huge relief for me to have Flash disabled as I do by default. And then, if I'm going to somewhere where Flash is the reason I'm going, then it's like, okay. You click on it, and it runs.

Leo: You just enable it there, yeah.

Steve: And I'll tell you, it's strange when you get used to your pages not being loaded with these really obnoxious sometimes Flash-animated ads, and then you go to someone else's machine that's running a generic browser where Flash is active, and stuff is jumping around, and frogs are coming out of the ads and all kinds of strange things are happening. It's like, oh, boy. Yeah, it's just better not to have it unless you want it.

Leo: And Web905 in our chatroom tells me, and I remember this now that he mentions the name, that the Firefox add-in that does the same thing is called Flashblock, one word. It's just like NoScript for Flash, basically.

Steve: Exactly.

Leo: Yeah. And we know how you feel about that. Well, I mean, this is one I would run. I think this is, you know, Flash is kind of a blight on the web, to be honest.

Steve: It's gotten carried away. It's like now there's escalation of who can make the most annoying, visually attention-grabbing ads. And, I mean, somebody should look at the page, and you hope you don't have epilepsy because this thing might trigger a seizure. It's just nuts how far it's gone.

Leo: There's a move afoot. One of the reasons Apple doesn't like Flash is because it's owned by Adobe. And there's a move afoot to move to an open standard that will allow this kind of animation in video and so forth in HTML 5 using vector graphics, SVG.

Steve: Scalable vector graphics, yup.

Leo: And you can do everything. I mean, I've even seen demos on YouTube's site where they don't use any Flash, and it's just as good. In fact, it's better. It's just that the browsers have to come along and support HTML 5. As they are rapidly doing so. I don't know if it'll be any more secure, though.

Steve: [Laughable] Well, we'll have...

Leo: At least it'll be open. I mean...

Steve: There'll be problems in the SVG implementations until we get those bolted down, so...

Leo: But they'll be open, and people will be able to look at them, and I think that helps a little bit, than having just a kind of opaque box. Question 4, Paul in Lancaster, PA wonders about custom apps versus commercial apps: Steve and Leo, the story of the Starbucks employee whose computer got infected by visiting a knitting site - he stuck to his knitting, and look what it got him - raised an interesting question for me. Is it better to write a web application like a forum or an online store from scratch, or to use an off-the-shelf system? Well, we know what Steve does, but anyway.

The way I see it, custom apps allow the programmer to put in just the level of functionality they want without having undue complexity. And as you say, Steve, complexity is the enemy of security. But off-the-shelf apps may be more secure because they're either open source, meaning there are theoretically more eyeballs looking for security holes, or they're commercial applications, meaning the company's reputation is at stake if they release an insecure application. I'm interested in hearing your opinion on this. Thanks for the great show. Yeah, he's clearly stated the pros and cons here.

Steve: Yeah, he has. And I think, I mean, he raises a good question. One of the liabilities of using a custom app is that when a problem is found in it, then the bad guys go looking for all the instances of it they can find because they've got an exploit that then they can multiply across all of the instances of the websites that are using that app. So that's a downside. In general, for most people, I think that the pro side of using custom, I mean, of using commercial apps probably wins because, as Paul says, you get the benefit of many people looking at it, of a company behind it whose reputation is, I mean, really stands to be tarnished if they make a mistake.

I would say you absolutely want to stay on the security upgrade train if you're using a commercial app. Make sure that you keep it current because we know for example many instances where commercial or open source code tends to get static on a server while the code is being moved forward and being made much more secure. If you're running something four years old, then you've got a huge number of holes accumulating that bad guys can take advantage of. My approach, obviously, as a serious coder, is I want to do my own. I want it to work the way I want it to. As he says, I want only the features that I want. You know, for example, my eCommerce system doesn't have the cart model because I just - or you don't have to sign up and subscribe and create an account. I find that kind of eCommerce site really annoying. If I imagine I'm just going to go somewhere once, I don't want to have to go through a whole bunch of rigmarole just to buy something. I want to buy the software and get out of there.

So I really do think that there are pros and cons. But in general, for typical programmers who are not serious major security-aware people, there is a chance you can - a very good chance because being secure is so difficult, writing secure code is so difficult, there's so many ways that you can be caught out, that if there was ever someone who really wanted to penetrate your site and focus just on your site, the exposure that you have, I think, if you do something yourself is much greater than if you use a commercial solution. But then promise yourself that you're going to keep it current, and really make the time somehow to keep the code on your server current. That's so important.

Leo: Good answer. Question 6. Lex Thomas in Research Triangle Park, North Carolina, is reminded of a programmer's adage. We've got to do more of those security sayings, by the way.

Steve: The security maxims.

Leo: Yeah.

Steve: Yup. We've got a bunch.

Leo: Good, yeah, someday - is reminded of a programmer's adage from 1984: While reading some articles talking about the just-released Windows 7 and the imminent Ubuntu release, I stumbled upon an old programmer's adage which was attributed to Datamation magazine. I remember that. Quote, "The activity of debugging, or removing bugs from a program ends when people get tired of doing it, not when the bugs are removed." Datamation, January 15, 1984. For those who are waiting for Microsoft to quit having Patch Tuesday, I'd say they are waiting for Godot.

Steve: It's funny, I love that quote. And I think it's so true. And it reminded me of something that I read that I've always really appreciated, that Donald Knuth wrote in the preface to a book of his. Now, of course we know Donald Knuth. He wrote the famous - I've got them behind me, you can probably see them in the video...

Leo: I see them, yeah, I recognize the binding.

Steve: ..."The Art of Computer Programming." Of course Don was the designer of Pascal, which was a language deliberately created for teaching programming and meant to really help convey the concept of block structuring and programming without go-tos and the notions of programming.

Leo: I thought Niklaus Wirth did Pascal.

Steve: Oh, gosh, what am I - of course.

Leo: He did MIX. Knuth wrote most of that book in a kind of a faux assembler called MIX.

Steve: Exactly, a pseudo assembly language, you're completely correct. I got the wrong author here.

Leo: Who could forget Niklaus Wirth?

Steve: Anyway, in his preface - and this is his book on Metafont, which is a huge program. Metafont is Knuth's typesetting system. And he said, "My goal in this work has been to write a computer program of which a professor of computer science might be proud" - which of course he is - "in spite of the fact that the program must meet real-world constraints and compromises. I've tried to explain thousands of details as well as possible, using the best documentation tools available. Since I have learned much in the past from reading other people's programs, I have also tried to make my own program sufficiently stimulating that it might give a bit of pleasure to its readers. There aren't many jokes, but several of the algorithms are amusing and/or amazing."

And then here's the point of this, which is what I loved. He said, "I believe" - now, this is a huge program, Metafont. I mean, it's big. He says, "I believe that the final bug in Metafont was discovered and removed on January 4, 1986. But if somehow an error still lurks in the code, I shall gladly pay a finder's fee of $5.12 to the first person who discovers it." And he said, "(This is twice the previous amount" - of course he's going in powers of two, so 512 cents - "and I plan to double it again in a year.) You see, I am really that confident." And I love that because it's his belief that this is bug-free code. But it took a professor of computer science who did nothing for years but carefully, carefully writing this one program to produce something that he believes, and apparently it's been pounded on substantially by a large number of people, to be absolutely bug-free.

Leo: It'll be the first program in history, however.

Steve: It's, well, it's because it is - we know that complexity is the enemy of security. And complexity is the source of so many bugs. It's just it's difficult to make a perfect large piece of code.

Leo: It's interesting that he asserts that it's perfect. That's kind of interesting. I never heard that.

Steve: Yeah.

Leo: Hmm.

Steve: We skipped #5, by the way.

Leo: Oh, well, let's go backwards. Thank you. I'm upside down. John in Baltimore, Maryland. Oh, yeah. Sorry. He's wondering about SSL certificate strength and key length: Steve, a recent discussion of SSL and man-in-the-middle attacks got me thinking when I needed to update my website with a new SSL web server certificate that uses 1024-bit key length. I noticed that both the Entrust Root CA and the Intermediate CAs use 2048-bit key lengths. My question involves the SSL certificate key length of 1024 used by many websites like Bank of America, PayPal, et cetera. Is 1024 adequate, and for how long, given the evolution of computer power? Is it time to consider 2048 bits for standard SSL certificates?

Steve: It's a great question, and it's something we've never really touched on before.

First of all, one of the confusing things is these key lengths, when we're used to talking about key lengths like 128 bits. The reason this is confusing is that these are public key lengths as opposed to symmetric cipher key lengths. The key lengths for symmetric ciphers, due to the nature of the way they work, are much shorter to offer an equivalent amount of strength. So, for example, today a 64-bit key length like DES, for example, a very old cipher - actually DES is 56 bits.

Leo: And an old broken cipher, at that.

Steve: Yes. Well, breakable. I mean, the block size is 64 bits, so it uses a 64-bit block. And that's now regarded as too few bits to encrypt at once because there just aren't that many combinations of 64 bits. Well, it's 2^64. But still, that allows you to, with modern-day computers and memory, to begin to build a table, even if it doesn't include the entire table, enough of it that you can begin to find collisions. So 64 bits is not a long enough block size, that is, enough bits to encrypt at once. Now we're at 128. And the key length of DES is a 56-bit key. Even 64 is regarded as, eh, we'd like it to be bigger. I mean, still that's a lot of - that's a large number of keys. But now 128 bits is considered a safe minimum for a symmetric cipher. But all the key lengths change when we talk about public key technology, that is to say, an asymmetric encryption and description where we use different keys to encrypt and decrypt.

Now, in John's question, he notices that typical SSL web server certificates are using a key length of 1024, 1024 bits. What's important here is that they're also all expiring within a couple years. That is, the keys that are issued by the root certificate authorities like VeriSign, Entrust and so forth, they all have expirations of one, two, and typically three years. I haven't seen any that are longer than three years. So it's that expiration length which allows them to get by with a 1024-bit key length because they know that no matter what happens, that certificate will expire within three years.

The reason the root certificate authorities themselves have doubled that key length, 2048 bits, is that the certificate authority generally has expirations way out in the future. I seem to remember 2038 and sometimes even further out than that. So their signing, their key needs to remain secure for decades. So as a consequence, just as an extra security measure, I mean, it might well be that 1024 would be enough for them. But they're saying, you know, we don't know what's going to happen between now and decades from now. So let's sign our certificates with a double-the-length 2048 bits because we are confident that that will allow us not to worry between now and the year 2038 or whenever their certs expire, sometimes even further out than that.

Leo: And now that machines are so fast, it's not a heavy burden to have double the bits.

Steve: Exactly. I mean, the public key technology is a lot slower, but you don't have to do it very often.

Leo: Right. We got Tim Lemmon in Atlanta, Georgia. He tried to give Disney the knuckle. We were talking - it'll come back to you. Steve, my family went to Disney World three years ago. When entering any of the theme parks, we had to swipe our membership card and scan our index finger for entry. Disney thinks it's like the

Homeland Security or something. Then I heard one of your podcasts about the subject. I always said I'd try my knuckle if I ever went back. Remember somebody did this instead of this.

Steve: Yeah, we objected to the idea that just for something like going through a security kiosk at a theme park, that anybody would be getting your fingerprints.

Leo: Terrible.

Steve: I mean, that's personal biometric information.

Leo: Yeah. So give them the knuckle. Last week we tried it. The entrance wasn't busy, so I swiped my card, then firmly pressed my index finger knuckle on the glass plate. System didn't like it, so I tried again. Same response. Third time didn't work. By then one of the employees noticed I was "having difficulty" and came my way to help. So I gave in and scanned my actual fingerprint. The system let me through with no trouble. Based on these results I have to assume, one, the system's programmed well enough that it realized there was not an actual fingerprint to scan; two, the system had retained my original fingerprint scan on file from three years ago, and my knuckle wasn't even close; or both. I started to ask the Disney employees, but quickly realized they had no idea. Makes you wonder what's really happening with the fingerprint scans; doesn't it? Of course the point of that is to match, to make sure you're the same person.

Steve: Yes. Unfortunately, and this is annoying, it's very likely that they do have his original fingerprint, and they've retained it for three years and will probably retain it forever.

Leo: In some insecure database, stored in the basement of Disney Central.

Steve: With employees who are as clueless as the turnstile employees, who didn't realize what was going on, in charge of that data. I mean, it is a concern. The problem of using your knuckle is probably getting a knuckle match the next time you go through three years later.

Leo: You have to start with your knuckle.

Steve: You've got to, exactly. Never let them have anything but your knuckle.

Leo: Yeah, that's - the damage is done, in other words.

Steve: And remember which knuckle. Yeah, I mean, exactly. And I guess probably they're not wanting you to, like, share your card around with other people.

Leo: I'm sure that's the real reason.

Steve: Yes.

Leo: And, you know, come on.

Steve: So the only thing I could suggest is, if you wanted to use a finger, don't use the finger they recommend. Don't give them your index finger. That seems to be the most often scanned finger. Or don't give them your thumb. Give them your little finger, give them your pinky because the system will probably think, wow, this is a small guy. But, you know, who cares what the system thinks. And but my sense is, from all the fingerprints that I've seen, a knuckle looks pretty much convincing unless you had some, I mean, it wouldn't fool a human. But I would imagine, if you'd always given it your knuckle from the beginning, it would probably say, okay, this is the same knuckle we saw before.

Leo: It's a weird fingerprint, but I'll take it.

Steve: Let the guy through, yeah.

Leo: Let him through. Or use your pinky or something. Something that won't be of use to anybody. Eric, reporting from an undisclosed location, asks about port knocking, SSH security, and Security Now!: Steve, your show is great. I've been listening since you first started putting out the series. I'm so glad you've been doing it. Thank you. I'm at a university and recently had someone hack into my SSH server. Ooh, that hurts. I was using freeSSHd for Windows XP. It was very strange in that it didn't appear to be someone who got in by brute force. Oh, even worse. It was a first attempt from a particular IP address, and there was no password or username entered into the log, just a connection attempt, and boom, there they were, connected.

Since then I've been working to reformat the drive in case they put a keystroke logger or other malicious software on it, and put up more secure SSH infrastructure. But it'd be great if you could spend a few minutes doing a roundup of SSH best practices, specifically something you spoke about many years ago, port knocking. Is it a combination of stealth mode, port knocking, moving the SSH port to something other than 22, et cetera? What is it you recommend? Many thanks. Eric.

Steve: Well. First of all, if someone gets into your system, then you can't trust your logs.

Leo: Right.

Steve: That's one of the first things that forensic security guys know is that your logs are useless, unfortunately, if your system has been hacked.

Leo: The first thing a hacker does is mess with the logs.

Steve: Exactly.

Leo: Cover their tracks.

Steve: Exactly. The first thing they do is to clear the logs in order to prevent you from really understanding what it was that they did to get in. So there's really no reason to believe that, from the logs - you just can't trust anything. So from that point it's not clear whether they used a password or not, how many times they pounded on it. I will say that I've heard many people who have SSH running on port 22, the default port for SSH, that the amount of connection and brute-force attempts to get in is stunning. So this is something that is really happening on the Internet now, and increasing. It's going up over time.

So no matter what else you do, there's nothing I would recommend more than not using port 22. I mean, that's the default port, so it's the last place you want to run your SSH server. It's trivial to change the port to something else. So, I mean, absolutely do. Some people might say, well, isn't that just security through obscurity? It's like, well, this is an instance where you've got massive scanning going on across the Internet of port 22. So why leave yourself open as a sitting duck and allow someone to connect to you? Clearly you need a very high-strength username and password because what they're going to be doing is just running through a password dictionary, trying to think of anything they possibly can to get in. And in an unattended system, or a system that you're not watching, they can be sitting there, pounding on your SSH server, trying to get in.

So absolutely put the port somewhere else. You'll notice, if you watch your logs, all connection attempts disappear. And so, sure, someone could scan all the ports at your IP, find a TCP connection being accepted there, and pound away on it. But they don't know what TCP connection it's going to be. And the probability of that happening is vastly lower. So it certainly makes sense not to run a known service at a known port unless you have to. You have to run web browsers at 80 because it's - well, practically - because it's really an annoying thing to tell users, oh, go to John's website, but then put :2637. And people go, huh? What? You know, the colon is the override where you tell you browser to connect, not to the default port of 80, or in the case of SSL of port 443, but rather to some random port that you specify. That's just not practical. But for your own SSH server, by all means move it somewhere else. Clearly, you want a username and password that will stand up to brute-force hacking. That goes without saying because you're inherently wanting to expose this to the Internet, which means here's an exposed service, you've got to protect it.

Now, port knocking we've talked about a couple times. It's a clever approach which allows you to use some other system of arriving packets at a given IP to open a port. Essentially, the idea would be, for example, you might try to connect to a certain port at the IP, then do a different port, then do a third port, then do a fourth port. And only after four attempts at specific ports in a specific sequence is then the actual port that you want to connect to made available.

The point is that connection attempts are TCP SYN packets arriving at that IP. So if you had something, for example, monitoring your router's log, the log would notice that you received a SYN packet. It would add an entry in the log that a SYN packet was received

from a certain IP. And then when you remotely tried to connect to a different port number, it would make that entry in the log. So if you had something watching the log, it could look to see whether a sequence of attempted connections to closed ports - these are ports that aren't even open. They're just dead. So they're not going to respond. But they'll still make an entry in the log.

So by deliberately connecting to a specific sequence over time, that's a way of keeping your port closed. And given that we've got 65535 ports, so that's one less than 16 bits' worth of ports, if you had a sequence, for example, of four different ports you had to try, that would be 2^16 times 2^16 times 2^16 times 2^16. Which ends up being 2^64 possible combinations, minus a tiny bit because you don't have a port 0. So that's a huge number of possible knocking sequences, just to get access to the service you're protecting, before you have access to it.

So it's a very interesting technology. I mean, port knocking is something, you can Google it, you'll find open source software. There are people who are using it to protect services. So, yeah. I mean, that's additional security if you've got a service exposed. And of course I did a little bit of Googling. I was wondering about just the overall security of freeSSHd. And just a couple weeks ago there was an announcement of a denial-of-service attack that's preauthentication. Which is to say that there is something people are doing which is crashing that service before they authenticate. Well, we know what that means. That means that maybe it's possible to do some sort of a buffer overrun before you've authenticated.

And in fact, if since that has happened someone has figured out how to crack through freeSSHd, it may be that it is in fact possible to bypass the whole username and login process using a buffer overrun which has not yet been widely disclosed. We may be right in the verge of, like, a new zero-day exploit for this freeSSHd daemon. So that's something you want to watch for, too. It's, again, another reason to get off of, no matter what you do, get off of port 22. Put it anywhere else.

Leo: Wow. I have to go change some SSH servers. I'll be right back. We're running them in Linux, though, and I think they're pretty hardened. But, hmm, it's good to know.

Steve: Yeah.

Leo: Question 10, our last question, and it is the Security Disaster of the Week.

Steve: Actually question 9 is our next one.

Leo: Oh, you're right. Then question 10.

Steve: And it's a special...

Leo: I'm all out of order.

Steve: We have a special treat for you, too, Leo.

Leo: Oh, good, okay.

Steve: It's there in red.

Leo: Oh, okay. How exciting. Let's start with Joe Dorward. He says he doesn't mind if I do this in Scottish. I might mind. Others might mind. He lives in Berkshire, England, and he realized that the free Internet access at the British Library may not be safe. Steve, I was at the British Library in London last week, and there were people everywhere with laptops. Very few had books open. And I realized they were only in the library to take advantage of the free WiFi Internet access. He gave us a long URL here.

Steve: Yeah. It doesn't matter.

Leo: Bl.uk/whatson/planyourvisit/wifi/wififree.html. A hacker's paradise, I thought smugly. They've no idea what they're opening themselves up to. Then I realized - I'm sorry.

Steve: You're really good at that, Leo.

Leo: No, I'm not. If you're a Scot you're going, oh, that's awful. It's like when you've heard Brits do American accents, and it's just painful to the ear.

Steve: Yeah.

Leo: That's what this is. Then I realized, in spite of listening to almost every episode of Security Now!, I understand the dangers of open WiFi hotspots, but I don't know how to take advantage of their free WiFi Internet access safely. So here's the question. Can you tell me what I have to do to use the British Library's free WiFi Internet access safely? Or is it just crap? No, I threw that in. Let's assume the people running the network know what they're doing, and there's a bad person already connected. How do we protect ourselves, Steve Gibson?

Steve: Well, we've talked about this a few times, so I apologize to our listeners who are going, oh, my god, we've already covered this. But it's an important issue.

Leo: It's worth doing.

Steve: Yes. So I just wanted to say, again, the threat model is that you want to be safe there, where you've got unencrypted connections. Open WiFi means that there is no encryption in the local hotspot. So that, because it's an Ethernet, inherently, anybody

with a laptop who has a modified WiFi adapter, which are easily found and available, can listen to all the traffic that is transacting there in the library. So anybody who is not otherwise secure is sending their email sometimes, well, many email passwords are not over secure connections. So standard POP and SMTP are nonencrypted connections. So anything that's not encrypted is going to be in the free and completely sniffable.

So really all you have to do is make sure that the stuff you don't want anyone to be able to access is over an SSL connection. SSL is your friend. It will protect you. So, for example, if you're using Yahoo! Mail, Hotmail, Google Mail, any sort of web transaction where you're concerned about security, make sure that you have a persistent SSL connection. If you want more than that, then this is where you need some sort of a VPN solution. We've talked about Hotspot VPN, which is OpenVPN based. And there are, you know, any kind of VPN, this is really what they're used for, is then all of your traffic will be encrypted from the time it leaves your computer until it gets to the VPN endpoint, wherever it is. Either a service on the Internet, maybe you have an endpoint running in your house, and so you link to your home, and then your traffic is decrypted there and goes out on the Internet in the clear.

So you just - you want somehow to have encryption active in that area where the danger exists, which is in this case where you're wireless until your data gets to the local hotspot that the library is running. And so if you can arrange that, then you can use their free WiFi Internet access safely.

Leo: And that's true of anywhere you are. Any hotspot, that's the way to do it.

Steve: Yes.

Leo: Finally - I'm sorry I got this out of order. I got it right now, I think. Eric Nichols in Odessa, Delaware, with the Security Disaster of the Week. Subject: FIOS WEP crack - say that three times fast - no packet sniffing necessary. I've been a long-time listener. I've heard a rumor that the ESSID of a FIOS access point is actually a packed version of the MAC address of its network interface. That's a good way of generating unique ESSIDs. However, they should have stopped there. It turns out that the default WEP - yes, WEP - key is [fanfare] the MAC address of its network interface. Doh. I found this website with a calculate to decode the MAC address from the ESSID - does use JavaScript, be warned - http://fioswepcalc.webs.com. For research purposes I tested it there. Yeah, worked on the first try. Pause for the collective groan. Keep up the great work. Say hi to Leo. Oh, my god.

Steve: Okay. So listen to what this means.

Leo: Oh, my god.

Steve: This is so bad. So they said, okay, we know that MAC addresses are going to be unique. So just to remind our listeners, a MAC address is a 48-bit thing which is 24 bits is assigned to the manufacturer and then 24 bits is incremented by the manufacture; so that all of their interfaces that they make, 24 bits' worth of Ethernet interfaces will all have, when concatenated with their manufacturer-assigned 24 bits, will have a unique 48-bit MAC address. That's important because packets are routed among the Ethernet

from one MAC address adapter to another. So you need to have unique addresses. And so this concatenation of the 48 bits is the way that was solved.

So then they said, oh, let's base the SSID, that is, our wireless access points beacon ID, on the MAC address. That way it'll be unique. And everyone said, okay, that sounds like a good thing. Unfortunately, they then, as Eric says, they took the MAC address and used it for the WEP key so that all of the Wireless Equivalent Privacy, WEP, the default WiFi, will have a different key, thinking that that was clever. The problem is, of course, what they're essentially doing is broadcasting the WEP key through the beacon, the WiFi SSID, which anyone is able to get. I mean, it's insane. You know, it's...

Leo: Well, talk about security through obscurity, they assume that you won't have sussed onto this, and so they're fooling you.

Steve: Right. And there's some packing going on so that you need a little JavaScript in order to undo this.

Leo: It can't be too complicated.

Steve: Well, no. And so this FiosWepCalc.webs.com, I went there. There's a little script that it runs. And so the idea is you put in the ESSID that you get from the access point, and it tells you what the WEP key is in order to connect to it, right through its security. Such as it is. Security Disaster of the Week.

Leo: Oh, my goodness.

Steve: Yes.

Leo: Steve, a pleasure once again. 10 questions good and true. If you want to watch us do the show live, we do it every Tuesday, I'm sorry, Wednesday at 11:00 a.m. Pacific. That's 2:00 p.m. Eastern time, 1800 UTC. Next week I'm going to ask if you can move, Steve, but I'll talk about that after we get off the air because we need to flip-flop with the Daily Giz Wiz.

Steve: We'll just have to tell John, since John Graham-Cumming is our guest next week.

Leo: Oh. I hope...

Steve: I think that makes it late for him. But I have no problem.

Leo: Well, let's see if we can do it Tuesday at 1:30 instead. And if not...

Steve: Oh, you mean like change days.

Leo: Yeah.

Steve: Oh, no problem at all.

Leo: And time, a little bit. Because the Giz Wiz, yeah, because Dick can't do it Tuesday, so he needs to do it Wednesday. So you can watch us do it live next Tuesday instead of our normal time Wednesdays. But, you know, forget the live, just download the podcast. You can get it from iTunes, Zune, anywhere podcasts are offered. And of course you can get it directly from Steve himself at his website, GRC.com, the Gibson Research Corporation. He has 16KB versions. He has transcriptions. He's got show notes. He's got the full version, too, all at GRC.com. That's also where you go to ask questions like the ones we just answered - GRC.com/feedback - and to find SpinRite, Steve's fantastic hard drive maintenance and recovery utility. Everybody should have it. If you've got a hard drive, you should have SpinRite.

Steve: And you know what the slogan is for SpinRite.

Leo: What is it?

Steve: It works.

Leo: That's simple. Short, but sweet. And it's true. Simple and true. You can't get better than that. GRC.com. That's where ShieldsUP! is, too, and all the other great free utilities Steve offers everybody to secure their systems. Next week, John Graham-Cumming. This is going to be fun. We're going to talk about JavaScript and why it sucks.

Steve: JavaScript Security, the oxymoron.

Leo: I love it. All right, Steve. We'll see you then.

Steve: Talk to you then, Leo, thanks.