Q&A #104 & The Firestorm

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-273.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-273-lq.mp3

Leo Laporte: This is Security Now! with Steve Gibson, Episode 273, recorded November 3rd, 2010: Your questions, Steve's answers #104, and The Firestorm.

It's time for Security Now!, the show that covers your security needs, watches out for you on the Internet, protects your privacy. And the guy who does it all for us, Mr. Steve Gibson of the Gibson Research Corporation, GRC.com. Steve, good to see you again.

Steve Gibson: And I guess we keep you on your toes in open WiFi coffee shops...

Leo: Oh, man, oh, man.

Steve: ...and similar places.

Leo: Yeah, the Firesheep adventure of last ep- if you did not listen to last week's episode, Episode 272, do, because that Firesheep - I installed it, you know, it was so easy to install. I put it on my Mac, on my little MacBook Air, brought it to the friendly neighborhood coffee shop. I didn't want to bring it, you know, it's illegal to use; right? I think it is.

Steve: Well, I don't know if it's illegal to view. It would definitely be illegal to double-click on anybody and acquire their credentials because then you really have intercepted their communication, like, proactively. But I would argue, if this is just being broadcast, as it is, then the broadcaster has some responsibility. And if you've got a radio that

receives what someone's broadcasting, and it shows it to you, it's like, well, okay, how is that wiretapping? So, I mean, it does get - there has been, well, first of all, the Firesheep has caused a firestorm, essentially...

Leo: Yes, yes.

Steve: ...of reaction. It's been - the developer was overwhelmed by the reaction from it. He didn't expect anything like it. At the moment we're at 571,600-plus downloads, so we crossed half a million some time ago. When we did the podcast one week ago, we were at 300,000, just a little over 300,000. So as you mentioned before we began recording, it's slowing down a little bit. But there's been all kinds of havoc as a result. And actually we'll be talking about a lot of that during this podcast today.

Leo: Yeah, I was - so I installed it very easily. In fact, I installed it at the coffee shop, it was so easy. First I wasn't getting a result. And then I realized, oh, I hadn't set the preference about which network card to use. Once I set that...

Steve: Yup.

Leo: ...I saw myself right away. But I didn't see anybody else. I guess nobody else in the coffee shop was using the WiFi. So I asked the friend I was sitting next to to fire up her Facebook, and she did. And immediately I saw her on the list. I said, watch this. I double-clicked it, and I was in her Facebook page. And I said, I could leave a status update. I could do everything but change your password. I could even change your privacy preferences right now. And then we called over the owner of the shop, explained the situation, showed her. And she immediately went upstairs and turned on WPA.

Steve: Oh, that's good. Well, and we ought to just say right at the top of the show, I loved the idea that you had for setting the SSID.

Leo: Oh, yeah, because of course the concern is now you're adding a password to your open access point, oh, no, what do we do? So I said, just change your SSID to "della," which is the original one, or maybe it was "bakery," I can't remember, and then in parentheses, "password is password," or whatever it is. Because you don't need to have a secure password, do you.

Steve: No, you don't. And that was the whole point is essentially everyone can know the password, but that still gives them individual encryption. Now, there are problems with that that we'll be talking about today.

Leo: Oh, good.

Steve: But I just thought it was very clever, Leo, that you put - you disclose the password in the name of the network because that's what comes up when you go to a

network for the first time. It lists them all. And if it said, like, "della (password is 'free')" or something, then you're, like, you don't even have to go to the management or have it posted anywhere. Everyone would know what it was. I just - that's a very - that's a neat idea, to put the password in the SSID.

Leo: So we will get an update, the Firestorm on Firesheep. And we also have questions and answers. All right, Steve. Let's I guess start with our regular updates.

Steve: Well, I did have a little comment. I got a tweet from someone named Ken Papp, who tweeted, following up from last week's episode, he said, "I did my good deeds today. Showed managers at Starbucks, CornerBakery, B&N" - which must be Barnes & Noble - "and Sheraton Firesheep in action."

Leo: Fantastic.

Steve: "Jaws hit the floor."

Leo: Yep.

Steve: And, see, and again, that's why I last week was jumping up and down about this. I mean, yes, I recognize that it creates a problem. But it's a problem that exists whether Firesheep is there to demonstrate it or not. And you saw the eyeballs on the manager at your bakery.

Leo: Oh, yeah.

Steve: You said, look, this is what is happening. And what I imagine will happen is this news will spread. There's, like, an initial adoption frenzy. But we have to know that, behind the scenes, organizations like Facebook are saying, whoa, we really need to step up our activities. I did receive a quote from them stating that they are working towards moving to pure SSL, and they are projecting that they're a few months away. And later on in the podcast we're going to talk about the costs of moving to SSL because there were some listener questioners about that. And I managed, actually I followed on from something, either you said it during the podcast last week, or I heard you mention it some other time, but about the cost that Google experienced.

Leo: They did a blog post, and I'll see if I can find that for you.

Steve: Oh, I have it.

Leo: Oh, you've got it, okay.

Steve: Yeah. So I tracked it down, and I got the actual numbers on Google's experience

in moving, for example, Gmail to 100 percent SSL.

So of course we've got the regular culprits in updates for the week. Firefox, remember that we talked about a just-discovered zero-day vulnerability that was used to hack people who went to the Nobel Peace Prize website. We mentioned it last week. And I knew at the time that they were working on an update. They have done that, and they've pushed it out. So what I found was I don't normally start my machine every day. It runs 24/7. And I leave, I mean, Firefox is like my portal to the world, so I leave it running. And that may be why I'm not getting updates all the time. But when I went under Help About, or Help Check for Updates, it said, would you like to install the downloaded update? So mine received it, but it didn't automatically, like, do its thing. So...

Leo: I think Chrome does it automatically. And that's one thing I like about Chrome is that it doesn't have to be restarted.

Steve: Oh, Chrome is stealth. In fact, I have a note here that, behind the scenes and silently as usual, with no fanfare, Chrome went up to v7.0.517.41.

Leo: I mean, that's a major, major jump. And they didn't say a thing.

Steve: Yup, just do-do-do-do-do-do, you know, all done. And the obligation, of course, is one of they'd really better not make a mistake because if they, I mean, they must have a...

Leo: I think you can turn it off.

Steve: ...a rollback. And they must have a rollback scenario where they're, like, able to recover from an update that damages the browser because then you wouldn't know, if you didn't know that you updated deliberately, or that it was just updated, you wouldn't know why it broke.

Leo: Yeah, what happened?

Steve: So, yeah. So Firefox is now at 3.6.12, 3.5.15; Thunderbird, because this was an across-the-board fix, two Thunderbird version chains, 3.1.6 and 3.0.10; and SeaMonkey at 2.0.10. So if you've got those, you've got the latest. And that fixes this, you know, emergency little update that our friends at Mozilla did.

Now, Shockwave did get fixed. Remember that we talked about it last - we talked about it on the podcast that I wanted to make sure people understood the difference between Flash and Shockwave, and that unless they knew they needed Shockwave, it may have been some debris that their machine collected over the last few years which, even though they weren't using it, was still creating a vulnerability because the nature of the attack was that your browser knows that it has access to the Shockwave plug-in. And so if you go to a malicious website that invokes that plug-in, the browser goes, oh, yeah, I've got that around here somewhere. Haven't fired it up for a few years. But oh, yeah, here you go. And then you're exploited. So it's a perfect example of something where, if

you don't know you need it, get rid of it because you're just better off.

If you do need it, though, on Friday after the podcast, Friday, October 29th, Adobe did update Shockwave to fix the problem. So it's now at v11.5.8.612 for both Windows and Mac. And again, you can, either way, if it's installed or if you want to update it, you can check by going to Adobe.com/shockwave/welcome. And if it gives you the whole animated song and dance, that's Shockwave Player doing that for you, which means it is currently installed. If instead it says, oh, you need a plug-in, well, this is where I would just say, back away from your computer. Do not click yes. And you're better off without it.

Leo: Yeah. I don't think there's much stuff anymore that uses Shockwave.

Steve: No, no. And it's - it was sort of the higher power platform that really didn't take hold.

Leo: It was mostly used in CD-ROMs. I mean, that's how old it is.

Steve: Yeah. Once again, we have a zero-day vulnerability hot off the press, actively being exploited in the wild, by our good friends at Adobe. And I want to ask them, how is that quarterly update cycle going for you? My goodness. They are scrambling because another vulnerability was found in Flash. The vulnerability is in Flash. But just like one we talked about earlier this year, I mean, not that long ago, it's invoked via the Reader app, and/or Acrobat, if that's what you're using for your reader. So pretty much across the board, everything is vulnerable. There's no fix for it now. This just happened, like, yesterday. Flash Player, well, the current version of Flash Player for Windows, Mac, Linux, and Solaris are vulnerable. The current Flash Player for Android is vulnerable. Notice that the Flash Player for iPad is not. But anyway, that's a different matter.

Leo: [Laughing] I got it.

Steve: And also Reader 9.4 and Acrobat 9.4 for Windows, Mac, UNIX and - yeah, Windows, Macintosh, and UNIX, those platforms are. The only thing that gets by is v8.x is not vulnerable. Adobe's acknowledged the problem. It's in our old friend the authplay.dll, which our listeners will remember us talking about last time there was a problem in authplay.dll. It's right now limited scope, targeted. Adobe is scrambling to address it. They have said that they will have it fixed for all those major OSes on the 4th of November. So we're recording this on the 3rd. The podcast normally comes out on the 4th. So maybe on the day people are listening to this it's available. So you'll want to make sure that you're running the latest version of Flash when you get the podcast. Android, the Flash Player for Android they're going to have pushed out and available on the 9th. And that's for the v10 versions of Flash. The v9 updates will be available on the 15th. And I won't rub it in to Adobe any more than I have that their next scheduled update was February 8th of 2011.

Leo: Okay. We can wait till then. No hurry.

Steve: Exactly. Well, and Adobe's not the only one in the doghouse. We also have a brand new zero-day vulnerability discovered just, again, just now by Symantec security researchers. It's been confirmed by Microsoft. It's in IE. It affects versions 6 and 7 and 8 of IE. But because IE8 has DEP, the Data Execution Protection, enabled by default, that is preventing data segments from being executable, IE8 is not obviously exploitable, or not nearly as easily exploitable. But it's still got the inherent problem in it. However, IE9 beta is not vulnerable.

What was found was emails being sent to companies with some questionable-looking language. I mean, I look at this, and it's like, okay, this is really not an English speaker who wrote this, we've all seen those, which contains a link to it. The link went to a well-known popular site. And I think it would have been a travel-related site based on the - it was blacked out in the sample that I saw. But the idea would have been it looked like it was travel arrangements and lodgings and things. And so right there the link was this well-known travel-related site where that would tend to give you a sense of, oh, I'm not really sure what this is about. I didn't know I was going on a trip. But I know about this travel-related site. So I'm going to click the link.

Well, the problem was, that site had been compromised with some malicious script which then checked the version of IE the user was running. And in the event that it was version 6 or 7, it bounced them to a site, I can't remember where it was, I want to say Poland I think is where this site was, which then that leveraged this brand new, previously unknown, zero-day vulnerability in IE which allowed a trojan to be installed. And they actually watched, they did a packet capture when they - the Symantec guys did, when they deliberately followed the link and saw what happened, and they watched someone connect to the trojan and then begin typing commands from wherever they were located into this machine. You wouldn't normally see it. It was going on in the network traffic, connected to essentially an invisible console that was running provided by the trojan.

So Microsoft's aware of it. I received by email, actually after I'd already put this together, in came a security alert noting that they were acknowledging this new problem. So where are we? We're still, it may not be time for second Tuesday of November. But so maybe they'll fix it by next Tuesday. Or maybe if it becomes a problem they'll fix it faster. Otherwise we may wait five weeks for it. But at this point no idea how to mitigate it; no idea of what to do to turn it off. If anything happens that's significant, I'll of course let everyone know next week.

There was, and I'm sure you probably saw this, Leo, a new iPhone lock screen bypass was discovered for iOS 4.1, that is, for iOS 4.1 phones. You tap, on a locked iPhone, you tap the Emergency Call button, then enter three pound signs...

Leo: Actually you can enter anything.

Steve: Oh, okay.

Leo: Yeah, you don't have to enter, you know, it just says you don't want to enter 911, obviously. So just any number of pound signs is good because it won't dial anything.

Steve: Ah. And then you hit the green Call button, and then immediately press the Lock button.

Leo: Yeah. The timing is tricky. So the first time it may not work. You've got to do it just right, but it's easy to do.

Steve: And then it unlocks the phone, and you have access to the phone, voicemail, call history.

Leo: Yeah. You don't get access to everything on the phone, just the phone app.

Steve: Right.

Leo: But that's enough to really screw with somebody.

Steve: Well, and I just wanted to let our listeners know, for the sake of their awareness, that if you're assuming that your phone lock is doing what you want...

Leo: It ain't.

Steve: You might hand it to somebody or leave it somewhere, and it's not.

Leo: Let's see if I can do this here. So you press Emergency Call. And it really doesn't matter what you dial; but, yeah, three pound signs is fine. And then the key is to press the green button, the Call button and the top button, the On/Off button, pretty much in a timely way.

Steve: Yeah. But what I read was hit the green Call button and immediately press the Lock button.

Leo: Yeah. And it just takes a little practice. But it's - whoops, I unlocked it. Let's lock it again, and slide to unlock, Emergency Call...

Steve: Sorry. What Apple said was that they were not going to do a patch.

Leo: You see, I got right into it.

Steve: It worked, huh?

Leo: Yup, yup.

Steve: That they were going to fix it in v4.2.

Leo: Well, and 4.2 is due soon.

Steve: Right.

Leo: So that's good news. But right now, without entering in the password, I was able to get into my phone app [indiscernible]. Isn't that nice.

Steve: Well, you've got skills, Leo.

Leo: Yeah, I've got mad skills.

Steve: Wait, wait, wait, did you just break the law? No, I'm sure it's not.

Leo: No, it's my phone. I think it's okay.

Steve: So India has now joined the UAE in announcing that they're dropping their promise or threat to ban Blackberry services. Which always was - we've talked about it before. I was a little uncomfortable about what was going on behind the scenes here. The direct quote was, "BlackBerry parent company Research in Motion (RIM) and the Ministry reached an interim agreement regarding government access to data sent over the BlackBerry network. RIM has promised a final proposal by January 31, 2011." And then what the UAE had said, similarly, was that "RIM had offered a workable solution."

What I, in digging around some more, I found an Indian newspaper online which had some details. And the concern had been raised that this interim solution wasn't very secure. Apparently what has happened is PCs have been installed in telecom providers, that is, so you've got whoever it is that's anchoring your BlackBerry wirelessly to the land. There's a PC installed there in which RIM installs some proprietary software which is able to reach out to RIM and contact them so that they perform the decryption. And then the decrypted data is sent back to that machine. So essentially it's a sanctioned man-in-the-middle architecture, which I got the sense that it was temporary, like they're going to come up with something. Everyone keeps talking about a formal or final agreement or resolution will be made by the end of January 2011.

So it does look like RIM came up with a technological means for weakening what was believed to be very strong endpoint-to-endpoint encryption because it was either that or succumb to the service being shut down throughout the region. So I was hoping that they were able just to say no, sorry, we can't do it, the technology won't let us, and we hope you won't disconnect us. But apparently the suits won, as suits do.

Leo: Suits have lawyers by the dozen.

Steve: Yeah, yeah. One other little bit of news I saw that I thought was interesting was that Amazon won a very important privacy battle with North Carolina tax collectors. North Carolina has been pursuing Amazon for some time, saying that we want complete

records of every purchase every North Carolina resident has made from 2003 through 2010, that is, through now.

Leo: They want to collect sales tax.

Steve: That's exactly what they want to do. And just I know you're aware of this. I wasn't until relatively recently, that the fact that we don't pay sales tax on goods that we purchase over the Internet from out of state doesn't relieve us the obligation...

Leo: Oh, no.

Steve: ...of paying them. It only relieves the sender the obligation of collecting them.

Leo: In fact, in California on your income tax return it asks you, is there any tax you want to give us? And you're legally bound to give it to them. If you don't declare it, you can go to jail.

Steve: Yeah. So we owe, we who are buying things on the Internet, it's called "use tax." It's not sales tax, it's use tax. It's the same, they just use a different word for it. But so, I mean, you can imagine the issue is that here's all these companies that are not taxing the customers; and they're saying, well, we can't possibly deal with all of the different state, county, city sales tax variations in order to do this.

And there's been a lot of effort in Congress to keep a taxation moratorium on the Internet under the interest of not doing anything to upset the apple cart of how nice eCommerce is growing, and the Internet's wonderful, and it's good for the economy and all that. Yet states are looking around and saying, wait a minute, look at all the money that we're not collecting because our residents we know are not reporting fairly the sales tax on the goods they're purchasing from out of state. Now, Amazon has no facility in North Carolina. So by law they don't have an obligation to collect that tax.

A U.S. District Judge Marsha Pechman in Washington state said that North Carolina's request went too far and, quote, "runs afoul of the First Amendment." So she granted Amazon a summary judgment on this suit. And it actually was Amazon who sued North Carolina, basically to buzz off and leave them alone. And Amazon stressed in its lawsuit that purchases - like, reminded everyone in North Carolina that purchases of books, DVDs, Blu-ray disks and other media enjoy special privacy protections. And so the reason I wanted to bring it up, not only is I think it's interesting relative to eCommerce, but this turns on issues of privacy because there's three different rulings and law that pertain. In a 2002 decision, the Colorado Supreme Court ruled that the First Amendment protects, quote, "an individual's fundamental right to purchase books anonymously, free from government interference."

Leo: Yes.

Steve: Yeah.

Leo: I love that.

Steve: Yes. And in this case the justices tossed out a subpoena from police to the Tattered Cover Bookstore requiring them to provide information about all the books that a specific customer had purchased. So they were trying to, for whatever reason, what books has this guy bought? Well, it turns out that a First Amendment right is that we can buy whatever books we want without the government finding out what they are.

And then in a 2007 case, federal prosecutors tried unsuccessfully to force Amazon to identify thousands of customers who bought books online, but abandoned the idea after a judge rebuked them. Judge Stephen Crocker in Wisconsin ruled that "the subpoena is troubling because it permits the government to peek into the reading habits of specific individuals without their prior knowledge or permission."

And then finally there's actually a formal law above and beyond what the First Amendment of the Constitution provides called the Video Privacy Protection Act, which makes it illegal for anyone selling - ILLEGAL for anyone selling movies to disclose customer information to anyone, including state tax collectors. The 1988 law specifically covers prerecorded videocassette tapes, but also sweeps in similar audiovisual material which of course would include DVDs and Blu-ray disks. So I thought that was just interesting info.

Leo: Of course as you I'm sure know, the Patriot Act does allow the federal government to find that stuff out.

Steve: Yeah. A little override.

Leo: Yeah, there's a little override there. Yeah, I got a letter from the state of California this year saying, we've decided not only do you have to pay use tax, but we want you to pay quarterly installments on your use tax. But it's interesting, now, can they, in this North Carolina judgment - not everything Amazon sells is books and movies.

Steve: Correct.

Leo: But I guess they can't pick and choose and say send us everything that's not books and movies. Or can they?

Steve: Well, it's not clear that this is over.

Leo: Oh, boy.

Steve: Basically, Amazon lost this one. But apparently there's also some - what had happened was, Amazon tried to satisfy them by providing the material anonymously, and in some sort of aggregate, like there's this much tax in this county and this much over

here.

Leo: But they want to know who.

Steve: Exactly.

Leo: C'mon. They can't - now, I have to say, in their defense, if you had a bookstore in the state of North Carolina or in California, you have to collect sales tax. You see it as a real competitive disadvantage to a company like Amazon that doesn't. And it's putting bookstores out of business. So there is an argument to be made that, you know, maybe Amazon should be collecting sales tax.

Steve: Well, and if - Amazon is Seattle-based; right? And if you are where Amazon is, then Amazon does have to collect sales tax for you. So that's a little weird, too, so...

Leo: Right, but that's normal, I think.

Steve: Yes.

Leo: If they do business in that state, then they have to collect sales tax.

Steve: Yes, exactly. Okay. So the cost of ubiquitous SSL. Following from a comment that you made, I did some research because I was curious what Google's experience was in switching themselves over to SSL because there are a number of different gotchas in doing this. And of course this all relates tangentially to the whole Firesheep issue. For example, why is it that Facebook is needing several months of exploration and work to turn on SSL? Like, what's the problem? Well, the one thing that I always hear is that there is a computational burden. And we've always sort of, I mean, it's like one of those urban legends that you just can't shake loose because it was once upon a time true, a long, long time ago, and it hasn't been for a long time.

So I want to just take the opportunity to dispel this with a little more data to back it up. The reason I have known it could not be true that there was a computational burden to SSL that was significant in this day and age is not only that, of course, processors are much faster than they used to be. You could argue, well, yes, but bandwidths are much higher, many more connections are coming in, so the load on the servers has scaled at the same pace that their power has increased and so forth. It's like, okay, fair enough. But what SSL acquired, and we talked about this in our podcast where we delved into the minutiae of the SSL protocol, is the ability to resume an existing session, that is, resume already negotiated credentials.

So what happens is the first time a client contacts the server, if the client already has in its own cache the credentials which are like, for example, less than 24 hours old for the domain, and like otherwise in every way qualify, when it's negotiating its handshake, it will provide the ID to the server of this negotiation that it already went through, which was the expensive part. Normally, if that isn't the case, there is a one-time cost to do the public key work to verify the signature on the certificate and to use public key encryption

in order to set up this negotiated credential. But that has to be done one time and doesn't need to be done for every other connection.

So SSL now has, and clients all are able to cache, that credential, which means that, for example, we'll take Facebook. When you initiate a connection, you're going to have to log on. So that's going to take you into an SSL connection where your username and password is exchanged, and you get the controversial unsecure key. And then the way Facebook currently operates, it takes you back down to a nonsecured, non-SSL connection where this cookie is transacted in the clear, which is of course the hook that Firesheep uses for being able to hijack or sidejack the session.

So the point is that client and server had to have a brief SSL connection anyway. So if Facebook instead kept the session, that is, all communications secure, then every time the client connected, as the user is clicking buttons and moving around the site and doing things, the client would be handing back to the server the ID that they had agreed on for this cached session, and there's zero overhead. No public key, no computationally expensive public key work needs to be done again. So the fact is it's just not expensive.

What I loved was I actually found a quote from the Google engineers who were responsible for moving Gmail over to pure SSL. And they wrote: "If there's one point that we want to communicate to the world, it's that SSL/TLS is not computationally expensive anymore. Ten years ago it might have been true, but it's just not the case anymore. You, too, can afford to enable HTTPS for your users."

Leo: I love it.

Steve: "In January this year (2010), Gmail switched to using HTTPS for everything by default. Previously it had been introduced as an option, but now all of our users use HTTPS to secure their email between their browsers and Google, all the time. In order to do this, we had to deploy no additional machines and no special hardware."

Leo: That's key.

Steve: "On our production front-end machines, SSL/TLS accounts for less than 1 percent of the CPU load, less than 10K of memory per connection, and less than 2 percent of network overhead."

Leo: Now, that can add up if you do a lot of transactions, though. That's not necessarily negligible.

Steve: Well, no, that's percent. So that's a fixed percent...

Leo: Yeah, but if you do a million, I mean, 10K is a lot of memory if you do a lot of transactions. I'm just saying, there is a cost.

Steve: Oh, sure. Sure, sure, sure. And it's not clear to me what the cost would be without SSL. That is, that 10K, that could be connection overhead.

Leo: Oh, maybe it's 8K, right, yeah.

Steve: So it may not just all be SSL overhead. Anyway, so they say, "Many people believe that SSL takes a lot of CPU time, and we hope the above numbers (public for the first time) will help to dispel that." Then they finally - and this blog post goes way on. But they said, "If you stop reading now, you only need to remember one thing: SSL/TLS is not computationally expensive anymore." So that was their message.

Leo: And they released that shortly after Firesheep came out. I mean, I think it was clearly a message to people like Facebook and Twitter: You don't have an economic excuse for not doing this.

Steve: Yes. And, you know, there was some flak that I received last week from people who didn't quite understand the whole notion of SSL everywhere, HTTPS ubiquitously, because they said, well, you know, but certificates are not free. And I have a web server, and why should I have to secure it? It's like, whoa, whoa, whoa, whoa. We're just talking about servers which have something to protect. You know, if you're running a web server, and you don't have an SSL certificate, then there's no security that your server is offering, and obviously therefore, I mean, hopefully, no need for it. Hopefully you're not doing anything over unsecured connections which would ever need to be secured because you're not able to take the person into SSL. So I was by no means meaning to say that HTTP needs to go away and be completely replaced by HTTPS. Not at all. I'm just saying that all of those scenarios which we talked about with Amazon and Facebook and MySpace and Twitter, in those scenarios it absolutely makes sense to switch to HTTPS. If you are ever using it, always use it, is the point.

Leo: Right.

Steve: So there's no - it's not like you're having to buy a certificate that you don't already have. You need one in order to allow people to logon securely. Well, you ought to also protect...

Leo: Just turn it on all the time.

Steve: ...everything else.

Leo: Yeah.

Steve: Exactly. Now, of course it does create some problems, for example, the famous mixed content problem. Browsers have always been warning us if some of the content of a page is secured and some isn't. That is, for example, if a secure page, an HTTPS page, asks for, like, even JPGs and GIFs and PNG images, which are HTTPS, then the browser will say, whoa, not all this page is coming securely. And it's like, oh, okay. It worries people.

It's not clear to me that it needs to, although you could design exploits that would take advantage of it. For example, if CSS or if chunks of script was being pulled, then while the page itself was secure, important pieces of the page could be pulled that were not secure. The reason I bring it up is that I can see a problem with third-party providers of content to a page, for example advertising sites. Advertising is almost exclusively nonsecure right now. And if you were going to switch your site over to SSL, exclusively to HTTPS, and you were going to be hosting ads, then you would need to be using advertisers that were also able to provide the ads over SSL. Otherwise your users would be getting these scary boxes saying not everything on this page is secure. And in this day and age where we want people to be more security aware, you'd like that to concern people.

So I can see Facebook needing some time to get their act together. I'm really glad that they are. And I think it's very clear. I mean, Amazon - actually, when I was talking to Mark Thompson about exactly this issue because he's setting up a very large and comprehensive web system for a deal that he's working on, he was talking about the issue of SSL. And I said, well, you know, turn it on all the time. And this was, like, months ago. I said, "Just have it on all the time. It's not expensive anymore. Just do it, and you'll never have to worry about it." And he said, "Well, I've been looking what other sites do." And he said, "I noted that Amazon doesn't normally keep you secure as you're poking around doing things. But anytime you do something important, like you want to check out your shopping cart, they require you to reauthenticate right then." And of course that is over SSL.

So Amazon's engineers, for whatever reason, decided it was worthwhile really moving people back out of SSL the rest of the time, and moving them into it - literally you're jumping back and forth as you do things on Amazon. It would be wonderful if one of these days we'd just see Amazon staying green up in the bar all the time, knowing that no snoopy people can see what I'm doing on Amazon. They may not be able to purchase something on my behalf, but they're able to watch where I wander around and what I'm looking at and what I'm searching for and so forth, which is no one's business.

Leo: Yeah, just turn it on. Costs nothing. Turn it on. Leave it on.

Steve: Exactly.

Leo: You've already got the cert.

Steve: And I did have a fun story, I always try to find something different, about SpinRite. Gary Harris wrote - the subject was "SpinRite Testimonial: Too Hot to Handle."

"Steve, I first crawled out from under a non-podcast-aware rock last Christmas, when my son bought me an iPod. I had no idea the wealth of tech information available until then. Where had I been? I immediately latched onto Security Now! podcast and promptly OD'd on the first 60 or so episodes until I finally caught up with the current flow sometime last February. I have been SpinRite-aware for many years, but I kind of forgot about it; and when I got out of the business for a while, I just had no need for it. While guzzling down your show episodes at an alarming rate, I decided to purchase SpinRite. I immediately went online, bought a copy, downloaded it, and then sat in wait for my first hard disk failure. I didn't have to wait long.

"I had a client who had a computer that was running much too slowly. I immediately expected spyware, viruses, yada yada, but I found no evidence of this. She said the computer had been running like this for at least a year. Before becoming SpinRite re-aware, I would probably have simply cleared the drive and reinstalled. But I decided to put the drive through my newly acquired 'spin cycle.'"

Leo: I love that, the "spin cycle." I like that.

Steve: The "spin cycle." "About 20 percent into a Level 2 check, SpinRite told me the drive was too hot to handle. It stopped scanning and let me know there was a problem. So I let the drive cool down and resumed the scan. It halted a few minutes later with the same message. I determined that, based on the client's statement about when it started, this drive had been in a state of near failure for probably a year, becoming slower and slower as it dealt with all the error conditions caused by its overheating.

"When I powered down the computer to remove the drive, it was literally too hot to handle. I had to let it cool. Since I had determined the slowness was due to the hard drive itself and not any malware residing on it, I was able to image this hot potato and apply the image to a new hard drive. The client called me a few days later and told me the computer was probably running 10 times faster than ever before. In reality, it was now running normally for the first time in a year." And he said, "SpinRite is, pardon the pun, a cool product. Thanks, thanks, thanks."

Leo: Yeah, I mean, I think when you look at something like the MacBook Air you really realize that the speed of your computer is often I/O bound. And something, a hard drive that's laboring, is going to really slow it down.

Steve: Yeah. I have noticed, when I've been purchasing state-of-the-art SATA drives, they really do seem to be running cooler than drives used to be. So I think we're getting heat under control. One of the things that SpinRite does is it polls the drive's smart data constantly while it's running. That allows it to do a whole bunch of cool things that nothing else has ever done, like monitoring the rate at which error correction is being used, monitor the health of the drive on the fly, and show you in a series of bar charts which are displayed while SpinRite's running. If these health parameters are being pushed down by using SpinRite, that shouldn't happen. And it's a really very sensitive, very cool early warning system that the drive's fine, but it's having more trouble than it ought to.

And one of the things SpinRite does is check the drive's temperature because of course we all know that people's fans or vents get clogged. Some people actually stick their computer in a closet, as if that's going to keep the machine cool. So many people have been surprised that the problems they were having were just that the drive was getting too hot. So although SpinRite was useful for keeping the drive in good shape, it also said, yeah, and do something about ventilation here, buddy.

Leo: Yeah. Now, are you ready, Steve?

Steve: Absolutely.

Leo: Questions, answers, Steve Gibson. He's on the hook, ladies and gentlemen. Of course he chose these questions and answers.

Steve: Is it on the hook or off the hook?

Leo: He is off the hook, that's true, too.

Steve: Okay, okay.

Leo: Question 1 comes to us from Todd Karwoski. Steve - actually, he tweeted this one. Thought you should know, Microsoft's Security Essentials wants to "protect" my computer from the Firesheep add-on. And he sent us a TwitPic of the message he's getting from Microsoft's Security Essentials. I'll give you a view of that here, and we'll zoom in. It says "HackTool:JS/Firesheep, alert level medium. This program has potentially unwanted behavior." So they think of it as malware.

Steve: So Microsoft was very quick to respond to this.

Leo: I think that's not unreasonable.

Steve: Okay, except that remember that a user would install this himself, or herself, on their own machine deliberately, thinking that it's cool. And then Microsoft Security Essentials is saying, wait a minute, this thing is potentially undesirable. So it's sort of - I guess it's protecting you from yourself, or protecting you from getting in trouble.

Leo: It doesn't say - it says it's medium threat. It doesn't say it's malware, exactly.

Steve: Well, see, I salute the fact that Mozilla chose not to block this.

Leo: Oh, interesting.

Steve: And they've got a blog posting where they discuss this. And they said, you know, this is not doing anything that the user who installs it doesn't expect. It does what the user does expect. It's open source. It's available. It's not taking advantage of exploits or flaws in Firefox. It's a utility. And we don't block utilities. So, and I think...

Leo: It is turning on raw sockets. I mean, are there some potential risks from using it?

Steve: I really can't see any risks to the user. It's something that the user wants. Now, the good news is that Microsoft Security Essentials does allow you to say, oh, thank you

for watching out for me. Spend your time...

Leo: It's more informational.

Steve: Spend your time patching Windows bugs, please.

Leo: Yes, by the way.

Steve: Yeah. It's just more informational. So I did get a kick out of the fact that MSE is saying, oh, wait, you've got Firesheep. Well, are you sure you want that? Uh, yeah, I'm having a lot of fun with it. Oh, and Leo, I didn't mention. College campuses just went insane...

Leo: Oh, I bet.

Steve: ...over Firesheep.

Leo: Well, and there is a risk that you'll be arrested for using it, so maybe that's what they're talking about.

Steve: Yeah.

Leo: I mean, if you misuse it, anyway. I wonder, do you get an error in MSE if you install things like Cain & Abel? Or Wireshark? Other hacking tools?

Steve: That's a good question. I don't know. I don't know.

Leo: I don't know, but I'm sure somebody will tell me. Paul Kawolski in Seattle, Washington with Question 2. He wonders whether WEP is enough. Steve and Leo, would using WEP encryption be sufficient to protect from Firesheep? We've mentioned that WPA works. In fact, somebody asked me, do you need WPA2 or just regular WPA? So clarify what will protect us from Firesheep, what won't?

Steve: Well, we've got a bunch of questions that sort of take us through that that are coming up. So what Paul was asking was about WEP. And sort of the answer is yes and no. Any encryption will protect you because Firesheep relies upon you using no encryption.

Leo: Unencrypted packets, otherwise it can't see what's going on.

Steve: Exactly. It's just passively sniffing the network. Now, the problem, however, with

WEP is that, unlike WPA, which does negotiate a per-connection encryption - and we'll be talking a little bit more about that, actually extensively more about that, here shortly. Remember that the way WEP works is that everybody on the same WEP access point uses the same WEP key, and that that allows you then to decrypt all of the traffic on the access point. So even though this is encrypted, it would protect you from Firesheep if the hacker did not have your password. But it would not protect you from Firesheep if the person running Firesheep did have your password.

So in other words, in a public WiFi hotspot scenario, we could never recommend turning on WEP encryption as a mitigation to the threat because it would provide none. In fact, Firesheep would work perfectly on a WEP-encrypted hotspot where everybody had the password in order to be on the hotspot because all the packets are encrypted identically and decrypted identically.

Leo: So there you go. Okay, there you go. So really WEP is not a good solution because that's - we're talking about turning it on in public access points; right?

Steve: Correct. And just to remind Paul that WEP is so badly broken now, in fact that was the title of one of our podcasts, that I think it takes 60 seconds to crack it. So it's just...

Leo: It's pretty easy.

Steve: Even if you didn't have the password, you could get in and then use Firesheep, yeah.

Leo: Chatroom is telling me that Microsoft Security Essentials does flag Cain & Abel, the hack tool Cain & Abel, but does not flag the packet-sniffing tool Wireshark. So there you go.

Steve: Oh, yeah. I've got Wireshark everywhere. And so I don't - yeah.

Leo: Yeah. Question 3 from Doug Johnson in Orem, Utah. He suggests using WPA might not help much. He says: In your last episode you mentioned that turning on WPA solves the problem of session hijacking via insecure cookies, as Firesheep does. This is a great step in the right direction, but it really isn't a cure-all. Even with WPA turned on, networks are still susceptible to ARP cache poisoning. Used in combination with Cain & Abel, Firesheep would still work even with WPA encryption turned on. A malicious user can send instructions to other computers to tell them to direct all Internet traffic to their machine, and that computer pretends to be the network's gateway, at which point client isolation becomes meaningless, and Firesheep is able to capture all packets from targeted computers, just as if the network was unencrypted and had no protection. The only real solution is for websites to force use of SSL for all authentication information. Is he talking about the cache poisoning that was a problem in the TKIP version of WPA?

Steve: No...

Leo: This is something else.

Steve: I want to, first, I want to acknowledge the last sentence of what Doug wrote: "The only real solution is for websites to force use of SSL for all authentication information." That's absolutely right, but we don't have that today. We're going to wait a few months. Maybe we'll get it from Facebook. Who knows what Amazon's going to do, and so forth. So I don't want anyone to misunderstand that I'm suggesting that turning on WPA solves the problem completely and forever, and that it's as good as or equivalent to individual websites really doing what they should be doing, which is protecting that credential which was negotiated during secure time, and they're spreading it around when you're not secure. So I agree with his last line.

But he's wrong about the use of Cain & Abel with WPA. And so this highlights a really good point, which is why WPA is so handy in this case. And it's what makes it different from WEP. With WEP, as I just said, all the packets are encrypted and decrypted with a single identical key. With WPA, each client negotiates their own - it's called a Pairwise Temporary Key, PTK, on the fly when they associate with the access point. When they connect up, they get their own pairwise temporary key, which is unique to their connection. And everybody's got their own, and that's what's used for [indiscernible] the AES cipher.

So what Cain & Abel does, and what ARP spoofing does, which we've talked about, is it allows you to essentially make yourself look like the base station to a user so that their traffic comes through you. The problem is, under WPA, even if you did that, and you can with WPA because it is an Ethernet protocol, which means there is a broadcast, there is the GTK, the Group Temporary Key, which is how broadcast packets are handled since there is a need to be running like Ethernet, and Ethernet has a notion of broadcasting, which is necessary for things like ARP to function because a station needs to say, hey, where's the gateway here? So it needs to send it out without knowing where it's going. It needs to just broadcast it out into, literally, into the ether, and allow the gateway to respond.

So WPA provides this Group Temporal Key as a means for ARP to function, which means that's exploitable by anyone using that access point. So that would allow them to redirect somebody else's traffic to them instead of to the gateway. The problem is, once the traffic gets to them, it's encrypted with their pairwise temporary key, which the attacker doesn't have. He's got his own. And so the only thing he can do, if he wanted to, would be to forward it on to the gateway, in which case he's really not a man in the middle, he's sort of a shuttle in the middle. He hasn't achieved any sort of interception at all at that point because WPA does provide that individual encryption keying per station, which makes it very useful as a mitigation against Firesheep. You turn on WPA, bang, Firesheep no longer works.

Leo: All right, moving along. You ready for more questions, Mr. Gibson?

Steve: You betcha.

Leo: We're going to talk about WPA-NoPSK mode for WiFi, the Pre-Shared Key. Steve, I just got done watching the Firesheep episode - our last episode. Since WPA

came out I always thought they should have an encrypted version of open access - oh, wouldn't that be a good idea - where you only have a secure connection to the access point, indicating with the SSID that it's open, something he would say called NoPSK mode. Personally, I think we need to completely ban open WiFi and WEP and possibly require user access control even in order to connect to them, as it is in fact very insecure, especially now that we have a Firesheep plug-in.

Now, I can't see any more vulnerabilities with having NoPSK than telling everyone the password or making it the same as the SSID. I think they need to add this to part of the standard - good luck with that, you know how long that takes - as we already have PSK and Enterprise 802.11x with a radius authentication. Why not this one?

I want to make one final comment that this does not protect us against man-in-the-middle attacks where the WiFi access point is impersonated. That's what we were talking about before; right? The only way you could prevent this is to use trusted certificates that are already installed on the machine, as is done with SSL in the first place. But I think it would still be a huge step forward in wireless security. Does that sound right?

Steve: Okay. So here's the problem. This is the final piece of this complex puzzle with WPA. If you don't have authentication - and authentication is the most, is the single critical thing that SSL and certificates provide. It's the authentication that we get when we receive the certificate from the server that we check versus our own list of certificate authorities, validating that the certificate we received is legitimate, and was hopefully issued by a responsible certificate authority, that allows us to believe that the connection we have we can trust. Without authentication, there is always nothing we can do about it, always a vulnerability to any kind of impersonation. If you think about it, there just isn't a way to avoid it. And WPA has that problem, even when it's encrypted.

So here's the vulnerability that exists if we tell open hotspots, like Starbucks, to turn on WPA. I say it's better than not. But here's why it's not perfect. Someone listening in while a new user is connecting is associating with the access point; will be able to see all of the traffic going in each direction. Which means that there isn't anything, fundamentally there isn't anything either the new user or the access point can know that the attacker cannot know. So if you have an attacker who's able to eavesdrop from the beginning, think about that. There's nothing either of the endpoints can know that the attacker doesn't because the attacker is equivalent to just being another user.

Leo: Right, it's a participant.

Steve: Yes. And so in practical terms, what that means is that the way WPA negotiates is that the access point invents a random number, generates a random number, and sends what's called the "anonce," a one-time use random number, to the access point. The access point, essentially it knows what the shared password is. So it generates a random number, its own, called "snonce," the station nonce. And it merges that with the anonce, the access point nonce, and the password, and then digitally signs that and then sends the access point back its snonce. So the access point, now, the access point knows the anonce that it sent, the snonce that was received, and it's able to verify the signature which was signed using the shared password, in order to verify that it wasn't interfered with in transit.

So the access point has the two random numbers, the one it generated and the one it got from the station. The station has the same two random numbers, the one it generated and the one it got from the station, which allows them then to generate from that the pairwise temporary key. And they're able to then encrypt everything under that. The problem is an eavesdropper saw the anonce go to the station and saw the snonce come back from the station of the access point. It can just assume that it was properly signed. But since it has the password also, it can even verify the signature. That is, if it's in from the very beginning of the conversation and is sniffing the traffic, although it has to do much more than Firesheep does, it's absolutely able to get that particular connection's key.

Now, an open access point is just unencrypted traffic flying everywhere. So, I mean, that's what Firesheep leverages. So turning on WPA I still say is very useful to do because you're bringing up encryption, and you're raising the bar much higher than it is now. There certainly could be Firesheep Pro, which is able to watch stations associate and obtain their per-station pairwise temporary key. You could even have a fancy one which deliberately goes in and disassociates a station, forcing it to reassociate, which would cause it to renegotiate a new key. That way you could get the key for even existing stations.

So having everyone using the same password has the liability that, yes, there is still a way around it. But that's, inherently, that's always going to be true unless you have authentication. And the only way to add authentication would be, for example, to give the access point a certificate which the station could check. And there we're talking about a huge change, way more than turning on a feature which all access points already have, but just have disabled by default.

So anyway, that's the one thing I wanted to say that I didn't get across last week, was that turning on WPA is not utterly absolute protection. You could have Firesheep Pro, which would be much more involved and a much higher level plug-in to create. But just turning on WPA shuts down Firesheep and brings up encryption, which I still think is a very worthwhile thing to do.

Leo: Couple more questions, starting with Iain Cheyne. Steve and Leo, want to share this very nice chart at DigitalSociety.org showing their analysis of their security under the Firesheep threat. Basically it's a report card for online services, red being no SSL authentication. And then they actually cover sidejacking and full hijacking...

Steve: Yeah, it's a really neat grid. I wanted to let our listeners know. There's a short little bit.ly link that we can just say it, it's bit.ly/cJ8Xig. And that'll take you to this much longer URL at DigitalSociety.org, which is a cool analysis they did in lieu of Firesheep. This was created in the wake of Firesheep to take a look at where major sites that are targets of Firesheep, like Amazon and Facebook and so forth, stand.

[http://www.digitalsociety.org/2010/11/online-services-security-report-card/]

Leo: POP mail is bad unless you have SSL turned on. SMTP, bad unless you have SSL turned on. IMAP, bad unless you have SSL. FTP bad. So turn on SSL on everything, obviously.

Steve: Yeah.

Leo: But a lot of these sites you don't have that as an option. And so that's the really - that's the issue, isn't it.

Steve: Exactly. And it's not under our control, as this next questioner asks.

Leo: Right. Moving along to our next question, Dean in North Dakota, he's asking about secure cookies. I'd like to hear about ways for individuals to enforce cookie encryption. Ah, there'd be a solution; right? One possibility is, no surprise, NoScript. In the NoScript options under the advanced tab is an HTTPS tab where one can enforce secure cookies. NoScript tries to append a ;Secure flag to cookies. I'd like to hear your advice on this or other solutions. Thanks for a great show.

Steve: Okay. So this is what I've talked about before. It is possible for a cookie to be, when it's issued, when it's given to the browser, to be flagged as secure, meaning essentially that tells the browser, send this cookie back to me, the server, every time you make a query to the domain which qualifies, but only if you're doing it over a secure connection. So the beauty of that is you don't have to worry about, for example, we've talked about SSL Strip, which is the man-in-the-middle attack that removes the S's from the HTTPSes, essentially turning an otherwise secure site into an insecure site.

In that case, even if the site was trying to be secure, but its cookies had been flagged as not secure, that is, had not been flagged as secure, then those cookies would be exposed any time a non-HTTPS query was made. If you simply flag the cookies secure, the browser - and all browsers support this, it's been there from the beginning - will never send a cookie unless the connection is known to be secure. So the problem is, if you turn that cookie, if you turn that flag on all cookies, then you would starve a site that wasn't expecting that of its cookies.

So the site would just be sending out the cookie. NoScript would force the cookie to be secure. But then, if the site took the browser back to nonsecure, the browser would say, oh, I can't send this cookie back because it's been flagged secure. It's because NoScript flagged it secure. The server didn't flag it secure. So the browser's not going to return it.

So it's of some value. It's the reason I never talked about it before is that I'm not really clear what this buys you. If you knew that you were always using HTTPS, then if that site wasn't flagging its cookies secure, by all means, absolutely, that's a cool thing for NoScript to do. But until we're always using HTTPS, it doesn't make sense. And if we're not using HTTPS, then we don't have security anyway. So it's kind of valuable, I mean, maybe useful. But I would say use it with caution because you could find it breaks things.

Leo: One last question. And this comes from Robert Walker in Atlanta, Georgia. He shares GitHub's post-Firesheep changes. Steve, I'm guessing you're already aware of this. But in case you haven't heard, check out the following. GitHub, by the way, is where Firesheep is stored. But it's used - don't think that it's somehow some spooky thing. It's used by programmers all the time. Even Apple uses it. It's a place, a code repository where shared code can be stored. Go ahead.

Steve: What I liked about this was that GitHub is one of the sites that Firesheep was aware of. And so they said, whoa...

Leo: Oh, I didn't know that. Oh, that's interesting.

Steve: Yeah. And so they said, whoa, we've got to fix this. And so what they did was they made some changes immediately afterwards where they did exactly what we were talking about with secure cookies. They flagged secure cookies, that is, essentially they created a session cookie for when you're not doing important things, and a secure cookie for when you are doing important things. So they used to have just one cookie which was used universally, whether you were over a secure connection or not. Now they've got two. And the one that is over a secure connection is flagged as being secure. So they have proactively made sure that they were going to be safe against this. And I think I misspoke, Leo. I meant the ToorCon, I meant that the ToorCon conference was supported by Firesheep. I don't know whether GitHub is now. But they said, well, we're not going to be caught out by this...

Leo: Just in case, yeah.

Steve: ...because we're hosting this. So we want to make sure we're safe against it.

Leo: Yeah, yeah. That probably is a good idea. And you don't want somebody checking in as you and changing your code. Actually has a huge security implication because a lot of code is stored there. And if you weren't aware of it, and somebody checked in and put a bug in your code, I mean, and by "bug" I mean a trojan in your code...

Steve: Yeah, it's open source.

Leo: Yeah, you'd have some problems. I'm going to save Tom's Cocoon and Bob Bent's question about NASes for another time because we're out of time.

Steve: Sounds great. We'll do them in two weeks.

Leo: You can, of course, always visit Steve's site, GRC.com, for 16KB and 64KB versions of this show. You can get the transcriptions there, so you can read along as Steve talks. And of course that's where SpinRite lives, the world's best hard drive maintenance utility, a must-have for anybody. If you've got a hard drive, you need SpinRite. Lots of free stuff there, too. GRC.com. Steve, we'll be back next week. Do we know what we're going to talk about? I think you had a topic, didn't you.

Steve: The Benchmark.

Leo: The DNS Benchmark.

Steve: The DNS Benchmark.

Leo: Great. We had to defer that last week for breaking news with Firesheep, and I'm glad you did. Couldn't have been a more topical topic.

Steve: That was perfect.

Leo: So thanks for your questions, folks. You can leave questions for Steve at GRC.com/feedback. GRC.com/feedback. And we record this show, normally, when we're not delayed by a Facebook announcement, every Wednesday at 11:00 a.m. Pacific, 2:00 p.m. Eastern time, 1800 UTC, at live.twit.tv. Actually next week, because we're going to switch back from daylight savings time, I think then we're going to be at 1900 UTC. I'll have to do my math later. I'll let you do the math. 11:00 a.m. Pacific time. And Steve, I thank you. We'll see you next time on Security Now!.

Steve: Thanks, Leo.

Leo: Bye bye.