Listener Feedback #101

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-267.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-267-lq.mp3

Leo Laporte: This is Security Now! with Steve Gibson, Episode 267, recorded September 22, 2010: Your questions, Steve's answers, #101.

It's time for Security Now!, the show that covers all your security needs from soup to nuts. And speaking of nuts, here he is, ladies and gentlemen, the star of our show, Steve Gibson from GRC.com, the creator of SpinRite, the world's best hard drive maintenance and recovery utility. Hey, Steven.

Steve Gibson: Hey, Leo. I'm a nut today, huh?

Leo: You're...

Steve: That's okay.

Leo: I'm a little nutty, actually.

Steve: Hey, if the adjective fits, wear it proudly.

Leo: We have a Q&A, Q&A 101.

Steve: Yes. Yeah, we have - not too much has happened in the last week. It's been a little quiet. There've been some interesting developments we'll talk about. And then

we've got interesting feedback from our customers, questions and thoughts. And actually we heard back from IT at San Luis Obispo about the issue of them giving their students a certificate authority. So that's mixed in there, too.

Leo: It's a state college; right? California State San Luis Obispo?

Steve: Yup.

Leo: CSULSUSOOOO...

Steve: Cal Poly.

Leo: Oh, it's Cal Poly. Oh. You'd think they'd have an IT policy that would work.

Steve: Well, they have a reason...

Leo: A reason.

Steve: ...why they're doing this. And we get it explained to us.

Leo: Excellent.

Steve: Not for spying on their students, as we were a little concerned.

Leo: And we have questions from you, our listeners. Why don't we - tell you what. Why don't we do the security news and updates, and then we'll get to our questions. And I have a Carbonite commercial.

Steve: I do have also a great piece of feedback from a SpinRite user. It's like, okay, this has been in my pile for a while. How have I not gotten to this one before now? So...

Leo: He, let me add, before - I want to ask you, I just got updated to the new Android 2.2 on my Droid X phone. And Motorola and/or Verizon shipped it out with Flash 10.1.

Steve: Yes.

Leo: I wonder if I should worry? Is Mobile Flash a different beast than desktop Flash?

Steve: You mean in terms of, like, being less of a concern?

Leo: Security, yeah.

Steve: Yeah, it's the same code base.

Leo: Oh, interesting. So it's the same potential problem.

Steve: Yeah.

Leo: Good to know. I will look for an update. There is an update, I understand.

Steve: Yes, in fact, one of our little blurbs of news here. In fact, it's the first one. So let's go right into it.

Leo: Let's head into it.

Steve: And that is that Flash was updated on Monday - two days ago from when we're recording this, three days ago from when the show normally airs on Thursday - which was ahead of its schedule. And that was to fix the zero-day vulnerability that we discussed last week. Both Flash and the Reader and Acrobat have brand new, relatively speaking, zero-day vulnerabilities. So Flash is taken care of.

And I did want to remind our listeners that we're sort of on PDF watch, or Adobe watch at the moment; that it will not be until the week of October 4th, which is week after next, that we get, per Adobe's promise - they're actually saying "the week of," so I don't even know if it's going to be Monday the 4th. But I'm sure they're working as hard on it as they can. And that'll be in advance of their normal quarterly update schedule. They'll be giving us new versions of Adobe Acrobat and Reader to fix a vulnerability that is currently being used for targeted attacks. So be careful with PDFs. Which is, I guess, always a good idea.

I was going to say that it's during these periods that we know there's a problem that you need to be careful. But the fact is, the nature of zero-day vulnerabilities, which seems to be what's happening with Adobe now all the time, is that you inherently don't know of a problem because it's being used before it's known of. Thus the term, so...

Leo: Thus zero.

Steve: Yeah, Microsoft had a big surprise also. There was a hacker conference last Friday in Buenos Aires. And it was disclosed during one of the presentations, and Microsoft has confirmed it on their website, that millions of Microsoft Server-based ASP.NET websites are vulnerable to a new attack.

Leo: Ugh.

Steve: Yeah. It turns out it's very clever. It's the kind of thing that it's like, okay, well, this is something they could fix. But they didn't see it coming. What some clever hackers figured out was that it was possible to probe a Microsoft ASP.NET server that's running millions of websites on the Internet now, it's possible to probe its crypto by sending it back ciphertext which the hacker makes up. And the nature of the error message which is returned...

Leo: Oh, interesting.

Steve: ...gives them a clue, a little bit of a clue about the way the server reacted to its attempt to decrypt their ciphertext, which they knew was wrong, but they sent it anyway.

Leo: Oh, interesting.

Steve: So you get different errors depending upon different styles of failure. And in aggregate, if you do that a lot, and you're smart, as hackers are, I mean, they're as smart as the developers are these days, they end up being able to crack the crypto and then being able to suck usernames and passwords and other contents out of Microsoft websites.

Leo: Holy moly.

Steve: I know, it's bad. So Microsoft is scurrying around. The workaround, there is a workaround which lots of server administrators are deploying right now, since this thing first went public, is to prevent different error screens. That is, Microsoft's formal statement, their formally declared, this is what you need to do, is you remove all variety of error and always return the same error, which of course means that the bad guys are no longer able to deduce things about the crypto on the server by looking at changes in errors. It's always the same error.

Leo: What kind of things could they deduce?

Steve: Well, I mean...

Leo: From how?

Steve: What I've read is that it is possible to get usernames and passwords from servers.

Leo: Really. Wow.

Steve: So that's not good. And it's a problem Microsoft's working to fix. The only solution they have at the moment is never return different errors. Force your server to always return the same error. And that way, even though technically the problem's still there, there's no information that's leaking back to the bad guys.

Leo: Right, right. Wow.

Steve: So it's like, whoo, yeah. Intel has confirmed the validity of the HDCP master key, which leaked out on the Internet, which we talked about last week. And so what this means is, this is not actual Blu-ray decryption, but this is - the HDCP is the sort of in-flight decryption. So, for example, satellites use it. And it's used for, like, from - it's used as the way the data moves through a digital system such that you want not to have the data in the clear at any point. So HDCP sort of is like, after the data is decrypted from the Blu-ray disk, then you use HDCP in order to move it through your digital system until it gets all the way out to the pixels, where arguably it's no longer digital.

So Intel is huffing and puffing and saying that they'll pursue anyone who develops anything. They're saying, well, you can't use software because software wouldn't have the performance that this needs. You have to do - people would require custom hardware. Well, custom hardware these days is an FPGA, a field-programmable gate array, which there's no question that there's 20 people, different people, all over the place, working on that right now, just to do it as a lark, just because now they can. Not that they're going to commercialize it, although I'm sure we'll see some commercialization of it in the Black Hat sector. But...

Leo: Why Intel? Why would they sue?

Steve: Oh, well, it's their intellectual property that's been breached. So they developed it. A subsidiary of Intel's developed it, has the licenses for it, licenses this to anyone who wants to use this HDCP protocol. And so it would be Intel that's saying, hey, this is our technology, we're going to go after anybody who exposes it. And I'm sure, I mean, the DMCA will just stomp them out of existence, if they can find them. Once again. And I'm sure you heard that Twitter had a major little goof that...

Leo: This is interesting. I couldn't wait to hear from you on this one.

Steve: Really interesting. It turns out that apparently a developer was using it to color - and I think it was a female, so I would say color her tweets on Twitter. What she realized was you could put a little JavaScript in a tweet; and Twitter, surprisingly, would not filter it.

Leo: Okay.

Steve: I know.

Leo: So I'm tweeting. I get 140 characters. I can put JavaScript in the text of my tweet. Not only would they not filter it, but the browser would act on it?

Steve: Yes, yes.

Leo: I thought the browser had to see, you know, bracket script bracket or something to act on JavaScript.

Steve: Yup. Turns out that the browsers are more permissive. What happened was that little hack was picked up by a 17-year-old Australian kid who's a few weeks short of graduating from high school, named Pearce Delphin. And he innocently used the JavaScript term "onMouseOver" in order to create a pop-up when anyone moused over the tweet that he had. He thought, oh, that's cool.

Leo: That's cool.

Steve: So he tweeted it. Well, it went out to everybody who's following him. And as tweets will, this thing spread like wildfire. Well, unfortunately it took no time at all, I think I read at one point that the vulnerability had a life of about five hours. But even in that window of time, bad guys figured out how they could reroute people to porn sites and create worms with this.

Leo: Oh, they jumped on it. They jumped on it, yeah.

Steve: Exactly. And it just went wild.

Leo: Now, what's interesting, and I don't know if this is still the case, but yesterday, if you do a search for "onMouseOver" on Twitter, you'll see a ton of tweets that are still there that have that content. Of course they don't work anymore. I guess they escaped it out or something, but...

Steve: Right. And Pearce said...

Leo: ...you could see what the code was.

Steve: Pearce said, "I did it merely to see if it could be done, that JavaScript really could be executed within a tweet," which surprised him as much as it would surprise anybody.

Leo: Well, yeah.

Steve: Just nutty. Just, it's like...

Leo: Now, if you are using NoScript, if you are a good Security Now! listener, you would be all right. In fact, if you use a third-party Twitter client, most of them were smart enough to escape out JavaScript.

Steve: True, although someone said that TweetDeck was not.

Leo: Really.

Steve: Yeah. So it wasn't the case that any or all third-party clients were safe because they were probably using, for whatever reason, the IE control in their client in order to render these things, and that would make them vulnerable to it. And of course anybody using the browser interface would have been.

And then in other very good security news, and I know you've heard this, Google has decided to start offering two-factor authentication.

Leo: It's so cool.

Steve: Yes. So for their Premiere, Education, and Government apps now, and then the standard edition stuff coming soon, you will be able to turn on cell phone loop, one-time password authentication. So that in order to sign into, for example, Gmail, you'll have to, at the time you sign in, Google will say, hold on a second, send us back the code we have just sent to your cell phone. And so you have to be in physical possession of your cell phone to receive that code and then enter it back into the form, which is waiting for it, and then you log in.

And there's been some criticism of this, saying, well, but how much security does that really provide? What if I got your cell phone? It's like, yeah, hold on a minute. Most of these breaches happen across the other side of the globe somewhere. Offline they're attacking usernames and passwords. So, yes, I would remind people we do not want to make perfect the enemy of the much, much, much better. This is much, much, much better.

Leo: Oh, yeah.

Steve: This is a great thing. And it's free. They're doing it. And positive commentary that I've seen have mentioned that here's Google doing free email login with a second factor, with useful second-factor authentication that is much more robust than most banks offer right now, for their financial transaction login. So anyway, props to Google. This is a nice step forward. This will give the concept some good exposure and begin to get users used to thinking like, hey, I can see how this is secure. Somebody has to have my phone - me - in order to log into something. Why isn't my bank doing that? Why don't...

Leo: Yeah. Isn't it great? Yup.

Steve: Yeah, it really is.

Leo: I immediately put the software on my Android phone that will generate the code. But unfortunately it hasn't been turned on in our Google Apps yet. But I hope so soon, yeah.

Steve: Cool. And I just wanted to make a comment about IE9.

Leo: Okay.

Steve: I'm not ready to talk about it yet because it's still pre-release and early beta. But I've been looking at it, and I've been reading about it. And there's a tendency that we have to - of inertia, of not recognizing when something that was originally really bad has gotten much better. And it really is the case that Internet Explorer, much as we had to move away from it because it was such a disaster for so long and deserved the reputation that it had, for the last couple major versions they really - Microsoft really has been making it better. And IE9 is another substantial step forward, just in terms of negotiating with its users for offering much better security.

I'm on Firefox. I love the ecosystem that Firefox has with the add-ons and the controls I have. I don't think that IE will ever be there because Firefox is fundamentally more knobs and switches and things that you control for tuning your experience just the way you want to. That really isn't IE's marketplace. IE is the browser that's just there in Windows and works. But I just did want to mention that it is substantially less awful from a security standpoint...

Leo: [Laughing] Faint praise.

Steve: ...than it used to be. But it needs to be said.

Leo: Sucks less.

Steve: It sucks a lot less, yes. It needs to be said. And I didn't know, this really wasn't an update or security news, but it's certainly errata. And that is that Firefox went just recently from 3.6.9 to 3.6.10, not to fix any security flaws, but because they were crashing on startup on some systems. So we got 3.6.9 on September 7th, and they spent the weekend figuring out what it was that they broke, and then they gave us 3.6.10 to fix what they broke. So that's good. And lastly...

Leo: Yes, I would say it was.

Steve: [Laughing] And lastly, everybody is all going crazy over this thing called "evercookie," the "evercookie." I'm getting it through Twitter. There was a ton of stuff in the mailbag about it. So I haven't had a chance yet to look at it because the news just broke. It's a researcher who has been experimenting with making, like, as the name sounds, the most absolutely super trackable, you can't shake this thing off no matter how hard you try, cookie ever imagined.

Leo: The evercookie.

Steve: The evercookie. And from what I've seen, he's been very clever with the stuff he's done. So I don't think it warrants a whole podcast episode because we have a strong underpinning now of understanding about the fundamental nature of this kind of thing. But I'll probably spend a little bit of time, maybe next week, talking about the specific things this guy has done in order to, I mean, just really drill down - of course it's all JavaScript based - but to really drill down into someone's PC and hook into it, hook onto it in a way that identifies you, unfortunately, pretty much no matter what you try.

Leo: Wow. I'd love to see the technologies.

Steve: Yeah.

Leo: I guess it's all JavaScript server-side...

Steve: Yeah, you need JavaScript, yeah, exactly. So, well, it's client-side, so it's JavaScript running on the client, which just does everything somebody who's been scratching their head and thinking, okay, what else can I get a hold of? And, I mean, bizarre things like specific colors in PNGs that are loaded to your machine or something. I mean, interesting sort of hacks. So we'll talk about it. I wanted to let everyone know that I'm aware of it, so to save your breath in tweeting and sending email. I know about it, and we'll be talking about it in some detail.

Leo: Excellent.

Steve: And I had a great piece of email from a Bill Morton, who - where's the subject here? The subject is, oh, "A legendary SpinRite story." He said: "Steve et. al, I first heard of SpinRite through the Security Now! and TWiT podcasts" - so right here - "and decided to have my workplace get a copy based upon the users' rave reviews about six months ago. Since then I've run SpinRite on just about every drive I've come into contact with and have been so blown away with its ability to recover drives, I felt obligated to submit my own review.

"Upon first getting SpinRite, I promptly went to my box of dead drives that I had acquired and fired it up. Amazingly, it was able to revive all but one of the most seriously damaged drives in my collection, which in that case had the click of death." Meaning it was offline, couldn't even be a drive. He said, "Next I ran SpinRite on all my hard drives and promptly found two brand new hard drives that had serious problems. I immediately backed up all my data, and Seagate replaced the drives, no questions asked."

Leo: Wow.

Steve: "But all of that was nothing compared to what I tried next. A laptop was brought to me that had serious disk errors, which left it both unable to boot and completely inaccessible to recover any data using another computer. As always, the data in question contained years of documents, photos, and music that had not been backed up and were irreplaceable. I set up a new drive and told" - he says, "I set up a new drive" - oh, he set up a new drive with that user. So he got him, like, gave the guy a new blank drive. "I set up a new drive and told the user that I would do everything possible to recover the data, but not to get his hopes up.

"Enter SpinRite. Right away, SpinRite began chewing away at the drive with the DynaStat recovery, trying to recover data from bad sectors on the hard drive. I could tell that this drive had serious problems compared to any other drive I had run. But I decided I would let SpinRite either repair the drive or witness the drive self-destruct. Most drives I run SpinRite on take several hours if they have no problems, and up to a day or two if they've got lots of bad sectors. This drive was at about 2 percent after 24 hours, and thousands of errors corrected. But curiosity as to whether the drive would catastrophically fail, or the seemingly unlikely event that SpinRite would finish, kept it running.

"I checked on the program daily, and after running for a week straight it was at 6 percent. From this point on, when I checked on the computer, I expected to see a pile of dust where the drive had once resided. But strangely, SpinRite continued to make tiny amounts of progress, so I kept letting it run. To make a long story short" - I think we're past that point here. Anyway, "To make a long story short, you can imagine that I was not surprised to find that one day SpinRite was no longer running. But I was shocked to see that the operations had finished. I anxiously checked the stats and found that SpinRite had run for 595 hours, 40 minutes, and 12 seconds, just shy of 25 straight days."

Leo: Wow.

Steve: "After taking pictures of the SpinRite screen for proof of this legendary runtime, I plugged the drive into my computer and was delighted to see that every single file was accessible. I promptly backed up the entire drive and made the user's day when I called to let him know that, despite losing all hope many weeks ago, I now had a full and complete backup of all his data. And that's a great feeling."

Leo: That's amazing.

Steve: "SpinRite saves the day again." And then he says, "You can see the photos of the recovery here," and then there's a link to SmugMug.com with a gallery link. So great...

Leo: Great. That is not a record by any means.

Steve: No. There have been people who have let it go for months, just out of curiosity.

Leo: Yeah. That's amazing that it was able to get - so what it's doing, and just for those, I mean, I think most people know. But what it's doing is it keeps trying to read that sector. And it doesn't time out. It just keeps trying.

Steve: Yeah. It does a lot. It's moving the heads around. It's able actually to recover data which is unrecoverable. I mean, the drive...

Leo: How would it do that?

Steve: Well, there's ways of reading the data in a raw format which takes it as it is. And then SpinRite's able to essentially apply its own algorithms, which is what the Dynastat data recovery technology does, taking a large database of up to 2,000 erroneous reads and then piece the data back together again. So there's a lot that it's doing. And as user after user finds, it actually does recover data. I just wish people would use it before their drives got in the condition. I mean, yeah, sure I wish, because then they'd buy it. But, I mean, it really can prevent drives from getting into this kind of condition. And that's, you know, we hear about these data recovery stories, but it really is fantastic preventative maintenance. I mean, I run it on my systems all the time, just, I mean, and so does Greg. He and I know more about SpinRite than anybody else on the planet, and we use it to keep these problems from ever happening. And I do get email from people, say hey, I bought a copy of SpinRite to support you. Thanks so much. I've never had a drive problem because I use it all the time.

Leo: Right. It will map out those bad or marginal sectors, so you don't have the problem down the road.

Steve: Well, yes. What it does is it shows the drive that there's a problem that the drive wouldn't otherwise find until it was too late. So it shows that there's a problem while the data is still definitely recoverable and correctable. But that scares the drive into - no, literally, it's like oh, crap, look how much recovery I had to do. We've got to swap in a spare sector, which is what SpinRite causes the drive to do. So it's funny because people say, yeah, I ran SpinRite, and nothing happened. It's like, no, unfortunately this is all transparent. So you can't see that it happened. But the proof is in the pudding that drives don't fail if you run SpinRite on them. So anyway.

Leo: It's actually good to say that because I think I didn't really understand that fully, so that's good. Are you ready, Mr. Gibson, for questions?

Steve: Absolutely.

Leo: Here we go. Question #1, as I scroll back to the top, Vincent Ragosta, Pittsburgh, PA has a follow-up question about Strict Transport Security (STS): After listening to the podcast on STS, I was wondering if this mechanism will prevent an SSLstrip attack from being successful. Thanks, and keep up the great work. What's an SSLstrip attack?

Steve: SSLstrip is something we talked about after one of the Black Hat conferences a couple years ago. Moxie, our old friend Moxie Marlinspike...

Leo: I love his name.

Steve: He demonstrated a man-in-the-middle attack, the kind of thing that could be effective at a Starbucks or anywhere like that where you've got unencrypted WiFi, where you are a malicious hacker who splices himself into someone's Internet connection, which is unfortunately trivial to do, and there are now automated tools that allow this through ARP spoofing where you just set up a computer in an unencrypted hotspot, and you route people's traffic through you.

What happens is, if you see them making a query with HTTPS on the fly, you remove the "S" and send it on. And if you see, when there's stuff coming back in the other direction, you look at the page that they're about to receive, and you take the "S" out of all the HTTPS so that they get a page with no security. Most users will assume that, for example, when they put in their username and password, that the site is taking responsibility for switching them into HTTPS. But that's done with text on the page, on the button that you press to submit the form. That's all there is, is just HTTPS, it says. So if Moxie or somebody who's a man in the middle removes the "S" as it's coming back to your browser, then your browser will submit the form unsecured, which allows everything that you submit to be captured by the bad guy.

So that's an SSLstrip attack, and it's not good. So the answer to Vincent's question is yes, Strict Transport Security, which forces your browser to use SSL, will not be fooled. That is, if your browser has the STS token from, for example, an eBay or a PayPal, saying only connect to me securely, the browser itself takes responsibility for adding the "S." So even if Moxie there was removing them on their way to you, your browser would say, wait a minute, this is eBay.com. I've got an STS token, so I'm going to just ignore the fact that there's no "S" on the HTTP and make an SSL connection. So, I mean, that's another really nice thing about Strict Transport Security. And it's another reason that we're hoping lots of people adopt it. Makes just total sense and really does solve a lot of these problems.

Leo: Question #2. I have another one for you, Mr. Gibson. Gerry Rachar in Victoria, BC, wonders about - I don't know what this is, maybe you do - Trusteer Rapport. Am I saying that right?

Steve: Yes.

Leo: Trusteer Rapport. Steve, I'm an IT professional and have a client who recently asked me what do I know about Trusteer Rapport? Which sounds like what's his name, Moxie Marlinspike. "Hi, I'm Trusteer Rapport." I have not give a reply yet. However, it looks like an additional product you install to block a keylogger when going to a protected site, mostly banks. When looking online I found a site in England that was producing movies showing successfully blocking keystroke loggers. I went to some sites that have videos of keyloggers getting through the protection. However, those videos have been taken down. It seems any dissent about this product is not taken well by the company. I could be wrong on this. I guess my

question is, don't you have to log keystrokes to block them? And what are they doing with this data? Does it work if a keylogger is already infecting the computer? Thanks for the show and the work. Trusteer Rapport.

Steve: Yeah. I've run across it a number of times. And I've seen listeners asking what it is and what I think about it. It's a commercial company, Trusteer is a commercial company that has this product that they call Trusteer Rapport. And an increasing number of banks, which is their main target customer, are recommending that users install this. It is anti-keylogger, anti-rootkit, anti-malware, essentially.

So the goal is that it hooks into the Windows API and prevents things like keyloggers from accessing your keystrokes as you log in. It also hooks in, for example, to the so-called WinINet libraries. That's the place where Internet Explorer and a few other Windows clients - interestingly, not Firefox because they bring their own SSL libraries. But a number of browsers that run on Windows use this library to perform their SSL, which means it's not being done inside the browser. It's being passed to the operating system.

What that means is that there's a shim, there's sort of like a place where you could insert yourself to capture the data from the browser before it's encrypted, which is one of the other things that some malware does. So what these guys do is they're in the business of producing a product to essentially enhance the security of Windows. Thus banks, among other vendors of sensitive information, are suggesting to people, hey, you might want to install this to improve the security of your online banking actions.

So it's a good company. It's not necessary, to answer one of Gerry's questions, it's not necessary to log keystrokes in order to prevent them from being intercepted. So their software is essentially trying to prevent malware from getting a hold in your computer. Now, as you'd expect, the malware is becoming aware of these people, and it's a cat-and-mouse game. So there are already some trojans that are Trusteer Rapport aware and remove these hooks that are blocking them from doing what they want. So it's the same old Spy vs. Spy, malware vs. antimalware battle, but one more useful tool for people who think it's a good thing.

Leo: All right. You give it your seal of approval. I guess if banks are using it, you don't have much choice. Right?

Steve: Well, banks are not - I don't think they're quite yet requiring it. But you can imagine that, I mean, there is some rumblings about them requiring it.

Leo: I can't imagine them requiring you run software on your system. That seems to me onerous.

Steve: That's a little much, yeah.

Leo: John Moehrke, who writes a Healthcare Security and Privacy blog - you say "blob," but I think you mean blog.

Steve: Oh [laughing].

Leo: He's a blob author. He wants to know something about OAuth terminology. If you did not hear our episode last week, Steve did a great job of talking about OAuth. Steve, I couldn't listen live, so I just finished listening to last week's. I'm now more confused than ever, he says. From a terminology standpoint, does OAuth provide authentication, authorization, permissions, or credentials? These terms were used too interchangeably for me to understand what it is that OAuth is doing. Sounds to me like it authorizes a service to impersonate a user. But it isn't clear to me how the service does this impersonation at a later time. You want to clarify?

Steve: So real quickly - I won't recap all of last week's episode because it's all there - what OAuth does is it provides a way, a means for a user to allow a third-party service to act on their behalf with, for example, a Twitter or a Flickr or whatever, Facebook, for example. So normally acting on your behalf or on the user's behalf would require that they had the user's credentials, which is typically a username and password. But you don't want to be handing those out. So what OAuth provides is a means for this third-party website to obtain limited credentials from that service, like Twitter or Facebook or whatever, without ever getting yours. So that's what it does. It does authorize the service to, in a limited way, impersonate the user, that is, act on the user's behalf, do some of the things that the user might want to do. It allows the user to authorize that without disclosing their own credentials to that third-party service.

Leo: Okay. So which is it? Authorization?

Steve: Well, I mean, unfortunately the terms "authentication," "authorization," "permission," and "credentials," those are almost synonymous. I mean, so you...

Leo: There's probably a technical...

Steve: You're providing authorization for that third-party service by authenticating with the primary service. And then it provides credentials to that third-party service, which the third-party service can then use for its authentication on your behalf.

Leo: Okay. That's all it is.

Steve: Yeah.

Leo: Yeah. Let's see, going to Question #6 - that can't be. Did I skip? Wait a minute, no, I skipped some.

Steve: We skipped #2 actually, also.

Leo: Oh, all right. Let's go back to #2. Brett in South Africa - there you go, sorry about that - wonders about VeriSign VIP Token for iOS. Oh, that's that VeriSign card thing that we talked about.

Steve: Yeah.

Leo: Steve, I was wondering if you'd seen VeriSign's VIP Access application for the iPhone, iPod Touch, and now the iPad. I have. I actually have installed it. It's free, but is it the real deal? Is it compatible? I'd love to hear your comments. Thanks.

Steve: Yes. I wanted to bring it to everyone's attention.

Leo: Should have mentioned this, I'm sorry, yeah.

Steve: Yeah, well, and we've talked about the football many times, the famous VeriSign football with the six-digit code which changes every minute. And some of our astute listeners back then realized that the first digit was always incrementing, which is used by the VeriSign back end in order to sort of stay synchronized. So it's a time-based solution.

Leo: And they also have the credit card thing; right?

Steve: Yeah, now, the credit card, however, was not time-based because it needs to - it uses a little battery that's built right into it. It uses that little eInk approach. So there it's a sequential - it's a counter which is encrypted that produces that result. So I wanted to let our users know that there is a program that VeriSign produces called VIP Access which, as you said, Leo, it runs on all of the Apple iOS platforms, so the iPhone, the iPod Touch, and the iPad. It is, again, clock-based, so it's time-based, much like the football is. And it's another token. It's another really nice authentication token, much like a football.

Leo: Is this the same kind of thing Google is doing? Is Google's app calculating it independently, or is it receiving a token over the Internet?

Steve: You do not run an app with Google, so it's sending your phone a text...

Leo: It texts you a token, okay.

Steve: Yes, it's sending you a text message.

Leo: There is an app, though. I got it, it's on my...

Steve: So Google's solution is universal for all phones, whereas this is only for Apple iOS devices. But we've talked, for example, about how it's annoying if you leave your football at home and you need to be able to authenticate remotely. Well, now you can have a full functional, fully secure VIP token in your iPhone or your IPod Touch or your iPad. You're only able to have one per device. So if you wanted - if you had a phone and a pad, you would need separate tokens. You're not able to move the same token between devices.

Leo: So it's using some sort of unique identifier in the device to see it or something.

Steve: Yes, and they do that deliberately to prevent it from being lifted off of your device and used on someone else's. So it knows where it's living, and it will not run, your particular instance of it will not run anywhere else. So they did that for security. So the only problem would be if the service you were using, like PayPal, for example, limited how many different tokens you could simultaneously register with it. I don't know that they do that. Or, if they do, it's probably five or six, so it's probably plenty for you to have a couple instances on different iOS devices and still a little football or the VeriSign credit card and so forth.

Leo: See, I'm puzzled because there is this Google authenticator software here.

Steve: Okay.

Leo: Will generate verification codes that can be used to provide additional security when signing. So I think, I don't know, maybe Google's either, in this two-step verification, offering you text or letting you use a program running on your phone that would work similarly to the VeriSign VIP Access application.

Steve: Okay.

Leo: So I'll look into that and find out for you.

Steve: And knowing Google, it's probably JavaScript.

Leo: Oh.

Steve: Well, I mean, who knows how they've made it secure. I would trust them to have implemented it in a useful and secure way.

Leo: Right. Moving along, I now will go back to Question 5. Sorry about Question 2. John Fecko in Cape Coral, Florida wonders about client-side security. Steve, your discussion about OAuth last week was extremely helpful. It was also very relevant to me personally because I'm a beginning web developer. My current project will eventually involve an iPhone and Android app, as one must in this day and age.

These apps will need to retrieve information from my database. But after listening to your discussion last week, I'm not sure how I can do that securely. Any key that I put into my app is vulnerable. We were talking about the HDCP key, right, and the fact that it has to be seen in the clear at some point because it's on an app.

How can I do that securely? Any key that I put in my app is vulnerable. Ultimate security isn't possible. Is there a more secure option, maybe a key that programmatically changes, like a garage door opener? Thanks for all you do and making me want to curl up into the fetal position from time to time. I know how you feel. So that's a question. What if you are challenged with this problem of having the key in software? How do you solve that?

Steve: Yeah, it sounds like, I mean, John didn't explain too much about what his app is doing. He says it's - he understands client-side security is what he's looking for. He says, "These apps will need to retrieve information from my database." So it sounds like, whatever the apps are doing, he would - I don't know if he's selling access to his database separately from the app, or maybe buying the app entitles you to access from his database. But apparently he's got some value-add which is the access to his database. So he wants to protect that so that the apps are able to access the data. But, for example, a bad guy cannot tear into and reverse-engineer the app, and then gain access to his database, without using the app.

And, John, assume the fetal position. Unfortunately, and this is what we've talked about so many times, this is the fundamental problem with client-side security. This is the problem with why you can't protect Blu-ray decryption, is my favorite example, because it's recent, and people have Blu-ray players. Every player is able to decrypt a Blu-ray disk. So it's possible to reverse engineer it, no matter how much the copyright owners would like you not to.

Similarly, exactly as you worry, John, it is possible for someone to reverse-engineer your app, maybe even easier if it's an Android app. I'm not clear on that. But still, if your app can access your database, then your app has to be there and can be understood by somebody to do the same thing. There isn't a way to protect it. The only thing you could do would be to issue each app its own authentication; and, if you discovered a pattern of malicious use, then deauthenticate, deauthorize that one particular instance of the app which had been, like, gotten out of control, reverse-engineered and so forth.

So, I mean, there are just - there are some things that we, unfortunately, that we just cannot do. And preventing reverse engineering of client-side security is one of the things we just can't do. We're able to not have this problem, for example, with SSL connections to web servers because we cannot have physical access to the web server. The web server has its key, its certificate, its SSL certificate, which we're relying on. And it's only our lack of access to it that makes that secure, the fact that we could only get to the web server through the Internet while we're pulling up pages. If anyone had physical access to someone's web server, they could get the key. And then there would be no more security. So the fact that people have physical access to the client means they could get the key, no matter what you do.

Leo: Question #6 from Joe, listening somewhere in the U.S. It's another OAuth question. Let's just get them all out of the way right now. Why are people, he says, using OAuth on the desktop? Shouldn't desktop apps just work via some API of their

own to their own website? Then any use of OAuth ought to work the same way as on the web. For instance, the Seesmic app wouldn't need to store OAuth tokens locally if it just communicated with Seesmic.com. Then the OAuth token stored at Seesmic.com isn't susceptible to the desktop vulnerabilities you mention in the podcast; right? What am I missing here? So he's saying instead of - and this is that vulnerability you were talking about. Instead of Seesmic, the application, storing the keys, the OAuth tokens, the authentication, it should verify with the website, where it's safer.

Steve: Right.

Leo: So why not?

Steve: So he's noticing that, as I was saying during last week's OAuth, and as I just mentioned when we were talking about the issue of client-side security, that it's fundamentally secure. What OAuth was designed to do was to protect one website's use of OAuth as it accessed another website. And it's because you don't have access to those servers that they're able to keep their secrets to themselves and not expose it to the desktop. So he's saying, okay, have Seesmic, which is vulnerable if it uses OAuth directly because it can be reverse-engineered and its keys can be obtained, have Seesmic connect to its own website, where now you have that website security because you can't get to the web server, and it can talk to the main service provider in a safe way. That doesn't work because all you've done is you've moved the problem.

Now, the bad guy, all they have to do is pretend to be the Seesmic app, talking to the Seesmic website. So you still have the same problem. That is, it's true that you could not get the OAuth token directly. But you're using the Seesmic.com server on your behalf to do the same thing. So if Seesmic is able to post on your behalf, for example, post to Twitter or do Twitter things on your behalf, if some bad guy wanted to spam your account, rather than obtaining the OAuth token to do so, the Seesmic OAuth token, they would simply pretend to be the Seesmic client, contacting Seesmic.com, sending things just as if they were you. So the problem is it's not OAuth that's really vulnerable. It's the client. It's the desktop. We don't have control of the desktop. We're downloading software all the time from hopefully trusted sources. Many of us, I mean, all of us probably are running software written by random people. Me. A lot of people run software I write.

Leo: You're random, sure.

Steve: I'm random. I mean, but not big corporate enterprises, but well-meaning developers whom we trust because they seem like good people. We look at what they do. Other people think their software is great. So, I mean, we're fundamentally vulnerable in this whole experience. And there just isn't a way to change that. If we want the flexibility and the power of using software from random people, then we're taking some random chance.

Leo: And that's why certificates and all this stuff, the web of trust, all of these ways

of kind of saying I trust you, you trust me, so it's okay to trust him kind of a thing, exist. We've got to find some way of doing that.

Steve: Well, and look at the mistakes that well-meaning companies make, like Adobe. We know that Adobe doesn't want to have all these problems.

Leo: No, of course not.

Steve: They'd do anything not to have them. But doesn't help them very much.

Leo: I did a little research into the Google two-factor authentication. Matt Cutts blogs about it. And it is in fact both. They will allow you to use SMS. But they also, if you don't want to give Google your phone number, for instance, they have applications for Android, iPhone, and BlackBerry that do the same thing that the VeriSign application does, generate a code. You can even use a voice phone call. They'll call you, and a machine will read you your authentication number if you don't have text. And they even, and you'll like this, support one-time single-use codes you can print out and put in your wallet.

Steve: Nice.

Leo: Just like you do.

Steve: Yup.

Leo: Like your Perfect Paper Passwords.

Steve: Right.

Leo: And LastPass does that. And here's the really good news. They do say that they're going to offer it, not just on Apps. I still don't have it on our Apps account here at TWiT. But they're going to not only roll it out on all apps, but they're going to roll it out to Gmail for everybody at some point, like in the next few months. And I think that's fantastic. You do not want to lose your Google account. It happened to Colleen, and it's a horrible thing. So I hope everybody will turn that on.

Steve: That's great.

Leo: Yeah. Question 78, Dan Malone, California Polytechnic State University. Cal Poly responds to our episode 265, a couple episodes back, Question 3. Steve and Leo, I'm responding to Brandon - Brandon was a student at Cal Poly.

Steve: Right.

Leo: And Brandon wanted to know whether or not his college is spying on him with this man-in-the-middle thing, certificate. He says: I work for the central Information Technology Services (ITS) organization for Cal Poly, California Polytechnic State University at San Luis Obispo, and I've been listening and/or viewing since the TechTV days. Hello there, Dan. Good to have you.

I was surprised when I heard about the Cal Poly requirement to install a custom certificate. After I searched around a little I found the reference to installing a root CA certificate on our campus residential network website, ResNet. As you surmised, the root CA certificate route was a benign choice, in this case both for cost savings and technical reasons. In the ResNet network, the residential network, Cisco Clean Access (CCA) is used for, among other things, authenticating network access. Since credentials are sent to the CCA appliances, they need to be protected with SSL. This is where the cost was an issue because there are a lot of CCA appliances. We've got 6,000 students living in on-campus housing, each requiring its own certificate, each of these CCA devices.

The technical issue has to do with the certificate format required by the CCA appliance and the difficulty of converting the format of previously purchased SSL certificates. So even though this cannot meet your Trust No One (TNO) model, I can say that the root certificate is used only for creating SSL certificates for the CCA appliances and a few ResNet support websites. So that makes a lot of sense.

Steve: Yes.

Leo: That's like what we were talking about with routers here that have their, you know, you create a self-signed certificate for the router, and then you have SSL to the router, that kind of thing. Cal Poly is using other technology for bandwidth shaping. More details are online at resnet.calpoly.edu/networksecinfo.html. These methods do not include decryption of SSL traffic. We do understand the concerns with installing the root CA certificate from a user perspective, and we'll work together to resolve the issues and move way from that model. We discussed implementing the changes over the weekend. Wow, that's quick response. However, the timing was not good. 6,000-plus students were moving in. School was beginning, so we agreed instead to implement the changes during the fall term.

On the other hand, timing was good since now the ResNet staff know of the issue and our plan to address it and can now answer questions from our many tech-savvy students and parents who may be listening to Security Now!. We probably brewed a little tempest in that teapot. We'll be meeting this week to go over options that will provide significant cost savings. Cal Poly and the California State University system are members of the Internet2 Alliance (that's internet2.edu) and the InCommon Federation (incommonfederation.org). The InCommon Federation now offers members unlimited server and personal certificates for a flat rate. Wow, that's a great deal. Cost of purchasing SSL certificates is no longer an issue, thanks to that.

To address the technical issues, we'll work with our central Network Administration group to see how they resolved the issues with the CCA appliances they use for our wireless network. We're doing a lot of great work with identity management in the

InCommon Federation, perhaps the topic of a future Security Now!? I'd be more than willing to discuss these further with you. Thanks for the show. Dan Malone, Lead Identity Management Architect, Information Technology Services, Cal Poly. Can you imagine the complexity of what they do?

Steve: Ugh. Well, and so there's a way to simplify this, that is, imagine if they had a thousand web servers and each one had a different URL, which meant that each one, that is, each of these web servers was in a different domain, which meant that each of the web servers would need its own SSL certificate. Which, if you didn't install a root CA in the students' browsers, that is, if you only relied upon the CAs, the Certificate Authority certificates that came with the browsers, that would require that some standard certificate authority sign these thousand certificates. Which we know they charge hundreds of dollars each for, with no quantity discount.

So that's a way of understanding the situation that Cal Poly found itself in. It's as if they had thousands of web servers, every single one of them needing its own certificate. I mean, I bitch and moan when GRC.com has to be renewed every three years because it's expensive. You can imagine the situation they're in where, due to the architecture of this Cisco technology, it's necessary for them to be able - for each of these devices to have a certificate.

So the cool thing is they are going to work on a way to get around this requirement and be able to get authenticated certificates for these devices that aren't going to cost them an arm and a leg. And then the requirement of putting their own CA into their students' browsers will go away. But that explains why they're doing what they are. It's got nothing to do with filtering their students' traffic, which is really nice to know.

Leo: Yeah, he explained it well. And I think it's amazing that they have a lead identity management architect in their ITS department.

Steve: And he listens to this podcast.

Leo: I would, if I were the lead identity management architect. #8, Steve in State College, PA really gets it about Net Nanny: Guys, during the last Q&A you mentioned that Net Nanny installs itself as a root certificate authority. Oh, here we go again. This begs the question, does the software generate a different root certificate for each user of the software? It seems much more likely that they just install their own common CA cert that's bundled in with their software.

But if everyone's sharing the same one, wouldn't it be possible for a malicious party to install Net Nanny on their own computer, capture the certificates it delivers in their browser when they browse to a well-known site, let's say BofA, BankofAmerica.com, and then reuse that generated data on phishing sites? To any user of Net Nanny there would be no certificate trust flags raised by such a site. And anyone else going there would see a certificate signed by The Net Nanny/ContentWatch root instead of a self-signed cert. Can he capture enough data to make that work?

Steve: Yeah. I mean, that's exactly - it's a perfect example of why you absolutely really

want to resist installing sort of off-brand CAs in your browser. We talked about this explosion in the number of certificate authority root certificates which our browsers are now trundling around with. And so the model of an exploit that Steve suggests here is exactly right. Assuming that all Net Nanny installations install the same root certificate, which is almost a certainty, then - so some bad guy gets Net Nanny and installs it on his own machine, then goes to BankofAmerica.com. And what his browser is going to receive in that transaction is essentially a fake BankofAmerica.com certificate, signed by Net Nanny, generated on the fly in his machine as he initiated that connection to the actual BankofAmerica.com.

But now he has a fraudulent, essentially, certificate for BankofAmerica.com, signed by Net Nanny. So if he then sets up a phishing site and, for example, uses DNS spoofing to cause some - or, well, not even DNS spoofing. Any kind of man-in-the-middle attack, for example, gets a whole bunch of these for a whole bunch of popular, secure sites, goes to a Starbucks, and intercepts people's traffic. He's now able to deliver fake certificates for all of these sites. Now, it's true that most people will not have a Net Nanny CA, Certificate Authority certificate, root certificate in their machine. But what they would get instead of, as he mentioned, instead of a self-signed certificate, if they looked at their certificate they'd see that it was signed by Net Nanny/ContentWatch and go, oh, well, I don't know what that is, but it sounds...

Leo: Must be okay.

Steve: ...important, yeah. And they'd click "okay." Whereas people who do have Net Nanny installed, and you can imagine maybe a targeted attack, where if you knew who Net Nanny users were and sent them links in email, you could then get them to go to a fake site that would raise no flags at all because their browsers, carrying that Net Nanny root certificate, would be quite happy with fraudulent certificates signed by Net Nanny, generated by the bad guy using the Net Nanny certificate on his own machine. I thought that was pretty clever. So, yup, that's a problem.

Leo: Wow. Geez, it's worse than I thought. JR Hallman in Ohio, inspired by the classic Portable Dog Killer episode: Dear Steve, after listening to #248 - and if you haven't heard it yet, please do listen, folks. It's not security, but it's a great story, the Portable Dog Killer episode - and with everyone talking about data encryption, I started thinking about making my own site for storing text encrypted. I know basic HTML and some PHP - I'm just starting to get a little nervous here - and how to make a complex site with a login system and data storage and encryption. So following your example, I opened up Firefox and went to W3Schools.com and started reading, then opened Notepad and started writing. I have been working on the site - I'm so scared. I've been working on the site almost from the time Security Now! 248 was made. I've been doing a lot of problem-solving to make it secure, and I've learned so much from the work I've done. Here's the site. Should we give it out?

Steve: Sure. It's really neat-looking.

Leo: Oh, good for him. CryptScript.com. It's not completely finished yet, but it's getting there. The server it's running on has six 10K RPM SCSI drives and RAID 10 for redundancy. Oh, he's 17.

Steve: Uh-huh.

Leo: This actually is a great story. Computers are my hobby. The first computer I had had 4MB of RAM, and the CPU I think was 75MHz and had a 400MB hard drive. It was running DOS; I installed Windows 3.1. He must have been, like, four. Seriously. That's, like, 15 years ago. Today my main desktop now has 8GB of DDR3 and an AMD Phenom II 955 running at 3.6GHz and 1.5TB of hard drive space, and I installed Windows 7 on it. I like to keep my computers very clean and secure, so I run almost everything in VMware unless it's something like World of Warcraft. He has a YouTube channel: youtube.com/v3dgames. Thanks for reading, and keep up the good work. So tell me, what is he doing here at CryptScript.com?

Steve: Well, he's just decided he wants to do something. And I just salute him as much as I could. He said he started out, he knew a little bit of HTML and a little bit of PHP, but NOT how to make a complex site with a login system and data storage and encryption and so forth.

Leo: So this is for his own purposes.

Steve: Well, yeah, he's just sort of - he says I want to, I mean, who knows what this will grow into. The point that we made during that show is, unless you do something, nothing happens.

Leo: Right. So he wrote it. And he learned all about it, writing it.

Steve: Yup. So he went to W3Schools, and he started reading, and he's learning about hashes and crypto. He's got a password generator there. He's got just all kinds of cool stuff that's online and that's working at CryptScript.com.

Leo: See, that's what I really like about this new digital era that we're in is that anybody who has an interest can find this information. It's widely available, freely available online. All the tools he's using are free. So he can go out and do it. He needs a computer, that's about it. Now, it looks like, oh, he's making - it looks like - did you look at his video site?

Steve: Yeah.

Leo: He's got the "12 Pains of Christmas" World of Warcraft theme. He's obviously a WoW player. And he's doing Machinima with Warcraft. That's really cute. Wow. [Laughing] Good for him. YouTube.com/v3dgames is his YouTube site. Yeah, this is a kid who's going to go far, obviously.

Steve: Yup. He's 17 years old, and he's not sitting around doing nothing. He's learning and stretching himself. And that's what it takes.

Leo: It does raise a point that we were talking about during that Episode 248 about the dog catcher. You worked in hardware. But nowadays kids don't need to even work in hardware. I mean, everything he's doing is software. It's all bits.

Steve: Yes.

Leo: But it's still building, it's still making, it's still creating.

Steve: Oh, yeah. And it's research, and it's sitting there with long hours and scratching your head and testing things and working on code. And, I mean, all of that grows you and makes you stronger in ways you just can't imagine.

Leo: Yeah, neat. Really, really neat.

Steve: Very.

Leo: Our last question, Steve. And it is our Adobe Flash Installation Tip of the Week. Timely, since you'll all be updating your Flash, I hope, if you haven't already done so. Steve, I think it was on one of your recent Security Now! podcasts I heard a discussion of installing Flash and how Adobe, annoyingly, forces the use of its own download manager. I discovered a way around this because it's something that's been annoying me for some time. I would have to either install the download manager or search for a direct link every time Adobe puts out an update.

Steve: Hm. Which is pretty often.

Leo: Yeah. But I found an easy way around this. All it takes is going to the Adobe site using a different browser. To update Flash for Firefox, you open IE, and you go to the Flash page (get.adobe.com/flash). You select the browser near the top. At the next page choose the correct operating system and continue. Select the radio button for the latest version of Flash Player for Windows - Other Browsers. That's the key, you have to select "Other Browsers." Click on the agree-and-install-now button, and an option to download the file instead of the download manager should pop up. The reason is, otherwise it would install it in the running browser.

Steve: Exactly.

Leo: So he says the opposite works for updating IE by using Firefox. You just have to select the IE version instead. Hope this is useful. Keep up the good work. And that has another advantage, which is you get to save that update file for later use, or use on multiple machines.

Steve: Exactly. I thought this was very clever. If it sees that you're wanting to update

the browser you're using, then it just does it, instantly sort of...

Leo: In the browser kind of, yeah.

Steve: Exactly. But if you come to it with a different browser and say, no, I don't want to update this type of browser, I want the other browser, then you get the file. So it's like, hey, I think that's very cool. That's what I'm going to do from now on.

Leo: Awesome. Steve, we've come to the end of our great questions from our great questioners. We thank them. If you want - now, we do this every other week.

Steve: Yup.

Leo: Do you know what you're going to talk about next week, or is it a surprise for us all?

Steve: It's a surprise for me.

Leo: We'll have something great next week. And then the following week, two weeks hence, we will have more questions. You can ask questions at Steve's site. GRC.com/feedback is the form. You also will find, once you get there, all sorts of stuff. Not only SpinRite, Steve's fantastic hard drive maintenance utility, a must-have for everybody with spinning hard drives, but you'll also find a lot of free stuff like ShieldsUP! and his Perfect Paper Passwords, the DNS Benchmark, which is still beta; right?

Steve: Yes, it's very close to being out in the world, very close.

Leo: Lots of good stuff there. Oh, and this show. 16KB versions, transcripts, show notes, all there: GRC.com. And if you want to watch us do this show, we do it live every Wednesday at 2:00 p.m. Eastern, 11:00 a.m. Pacific, that's 1800 UTC, at live.twit.tv. So you can tune in and watch live. We have a chatroom going in irc.twit.tv, which I watch during the show. And it's always useful to have you on the chatroom, so anybody - I think of them as my brains, my virtualized brains. It's irc.twit.tv. And we are now putting the show notes in our wiki. Actually, I think we've done this all along, but I've been a little bit more assiduous about getting them there. And the wiki is wiki.twit.tv, and then you can go to Security Now! show notes, and you'll see all the questions there, the show notes and so forth. Steve, thanks so much.

Steve: Always a pleasure, Leo. Talk to you next week.

Leo: See you next week on Security Now!.