The Fourth Factor

Description: Having discussed the first three "factors" in multifactor authentication (something you know, something you have, something you are), Steve and Leo explore aspects of the power and problems with the fourth factor, "someone you know."

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-094.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-094-lq.mp3

INTRO: Netcasts you love, from people you trust. This is TWiT.

Leo Laporte: Bandwidth for Security Now! is provided by AOL Radio at AOL.com/podcasting.

This is Security Now! with Steve Gibson, Episode 94 for May 31, 2007: The Fourth Factor.

It's time to talk security. And our good friend Steve Gibson is here from his Security Now! labs in Irvine, California, home of...

Steve Gibson: My lair of security.

Leo: Your lair, your secret cave, the bat cave, home of Security Now! and, of course, of SpinRite and SecurAble and Shoot The Messenger, DCOMbobulator, the famous 50 million-strong ShieldsUP!. Did you ever figure out why you got a bump in ShieldsUP! last week?

Steve: No, but it's still there. Knowing that we were going to talk about it maybe today I looked again a few minutes ago. And it's consistent. We're, like, 49,000 uses of ShieldsUP! a day now.

Leo: A day. Wow. Somebody's got to be plugging it. I guess it could be the radio show. But I wonder if maybe now a router's not saying, or some security software's not saying, "Test us."

Steve: I think that's probably exactly it. Somewhere, someone with a serious following is saying, you know, use ShieldsUP! to go check things. And but what's really interesting is, as I said before when I mentioned this, and I saw it again profoundly, is this weekly cycle. And it does look like it's actually slower on the weekends and busy - yeah, sorry, Leo, it's not the

radio show in this case - and busier, I mean, it's a substantial population of people who are now, like, I mean, it's a sharp peak. It's like something discovered it overnight. So I'm sure you're right. Somewhere there's some high-traffic site or facility which is saying, you know, use ShieldsUP! to test yourself. And...

[Talking simultaneously]

Leo: ...because we certainly have - we've mentioned it from time to time, but we've been harping on it a lot because of the 50 million. It could even just be the attention this podcast generates.

Steve: You saw the chart. In fact, why don't you bring the chart up right now? Go to GRC.com and go to ShieldsUP!, and then click on that first entry button to get into where it sort of shows, you know, it says, "Greetings," and then you see that first chart. You see the first set of buttons. And then scroll all the way down to the bottom of that to item no. 12, which I think is labeled "Site History." And if you click that, you'll see I build a little...

Leo: It's neat that you put that up. That's great.

Steve: Yeah, I build a little dynamic chart that shows. But, I mean, it's overnight. I mean, in one day...

Leo: Oh, yeah, it's very visible on this.

Steve: ...it doubles.

Leo: Because it's very consistent, too.

Steve: And you can see there used to be a sine wave, historically, which is our weekly, just the standard weekly cycle. And this is - so it's more pronounced now. The sine wave is a higher amplitude. But something happened in one day where suddenly and consistently now the traffic doubled.

Leo: Wow. It's very clear, yeah, when you look at that you know that something happened. Wow. Anyway, 50,597,626 shields tested.

Steve: And counting.

Leo: And counting. By the time I read it it's much, much higher. We want to, before we get into our subject, today we're going to kind of do a follow-up on our three-factor authentication discussion.

Steve: Yeah, actually I think of it more broadly than that, Leo. My feeling is that authentication is going to become nothing more than increasingly important in the future. I mean, as online banking happens, as our identities are - as it's more important for people's daily lives that they

be able to prove over the Internet that they are who they say they are, I mean, that the applications are expanding, and as that happens it becomes increasingly important to be able to prove you are who you say you are and for someone you're proving it to to be able to trust that proof. So I want to expand on and talk some more about some specific issues of authentication. But this is more than, you know, we certainly opened the topic broadly when we, four weeks ago, talked about multifactor authentication. Of course we started off the series 94 weeks ago talking about passwords, which is sort of...

Leo: Right, right, that's right.

Steve: ...the classic, easiest, and most familiar form of authentication. So I think our listeners should assume that authentication is going to be an ongoing sort of topic thread for us, probably for the foreseeable future.

Leo: It's critical. It's a critical subject. And Steve has discovered a fourth factor. So it's exciting stuff.

Steve: The Fourth Factor.

Leo: That's what I'm going to call this one. First of all, when you say "authentication," just to make this clear, what do you mean when you say "authentication"?

Steve: Well, it's about proving that you are who you say you are, authenticating that you are - that, well, and okay, authenticating that you are you, and doing so to somebody remotely. I mean, that's where the challenge comes in. Remember that when we talked four weeks ago about multifactor authentication, we started off talking, for example, in the olden days, I think we used some examples from characters from "Andy of Mayberry," where Opie would go into the drugstore, and he could be allowed to pick up Aunt Bee's medication because the druggist knew him, and so he knew it was Opie coming into the store, and that Opie was trustworthy and would go straight home and take Aunt Bee her pills. And so four weeks ago we talked about the three factors of authentication being something you know - meaning, for example, like a password; something you have...

Leo: Something you and only you know. It's private to you.

Steve: Good point. And in fact we drew the distinction then that we're talking about discriminating, that is, something you know that no one else knows, meaning that that factor can be used to discriminate you from anybody else, given that you've kept this information to yourself.

Leo: And that's exactly what a password does, I mean, it tells the website or your software that it's you and nobody else.

Steve: Right. And then of course the something you have is a physical token of some sort. And of course the popular one that a lot of our listeners, I was surprised how many people wrote in and said, hey, we're using SecurID, which is an RSA token which we've talked about now several times. It displays a periodically changing, multidigit number in a little LCD screen. And so it's the kind of thing you have on your keychain, and it's always showing some number, and

every minute it goes to a next one. And then by hooking that into an RSA server, the server knows the seed that your particular token is set with, and so it's able to determine what number you're seeing, so it's able to verify that, well, somebody has your token, and we're hoping it's you. And so it's funny because a number of our listeners who wrote in said, hey, not only do we have to give them that token number, we also have a PIN or a password. Apparently that's software configurable. Some companies use PINs, short numeric PINs. Others allow employees to use regular passwords of some length.

Leo: So that would be dual factor, so it's something you know and something you have.

Steve: Exactly. And that solves the problem of your token getting away from you and it being used for some mischief.

Leo: That's what a bank card does with an ATM card, exactly.

Steve: Exactly. And in fact there's another, much more common example of two-factor authentication. Now, the ATM card has the limitation that you have to be at an ATM machine. And that's why this RSA token is cool because, since it's an LCD display, and you're entering the data on a screen on a website, you're able to get hardware token two-factor authentication that way. Whereas the only way to do that, for example, in a PC format would be if you had a mag stripe reader hooked to your PC. Then you'd use your ATM card to do a card swipe.

Leo: And there are Smart Cards to log onto machines, like Power LogOn, that do exactly that.

Steve: Right. And then finally the third factor is something you are, which gets us into another large topic that we're going to be spending some time on in the future because I think it's fascinating and also really interesting. And that is, you know, things like the whole idea of biometrics, that is, something bio, that is about you, and metric, that is to say that can be measured. And so there's retinas and irises and hand geometry measurements, and of course very common is fingerprints.

It's funny, Leo, I've been using now for several weeks the fingerprint scanner on my IBM ThinkPad, and I really like it. It has support in the BIOS, and it uses the TPM, the Trusted Platform Module, built into the ThinkPad, which is another topic we're going to be talking about in the future because it's been controversial in the past due to the possibility of its being abused. But I really like the idea that my hard drive is locked so that no one who even took the drive out of the laptop would be able to read it until I swipe my finger as the laptop boots up. And the BIOS, an interaction between a processor in the fingerprint scanner and the BIOS authenticate me in a truly secure fashion using the Trusted Platform Module that's built onto the motherboard of the laptop. Only if that succeeds will it then dynamically unlock my drive as I boot to allow there to be any chance to boot an OS. And then Windows comes up knowing that it's me that has turned this laptop on. So it's a really nice solution. If I were more concerned, I could go for two-factor authentication and have to provide a password and my fingerprint. But I'm careful about my laptop, and I'm not traveling...

Leo: Let's not go crazy here.

Steve: ...blah blah blah, exactly.

Leo: Right, all right. So you've discovered, now that we know what authentication is and we've got the three different factors, you've discovered a fourth factor.

Steve: Well, yes. During the research I was doing four weeks ago I stumbled across something that just caught my eye and, you know, grabbed some references to it and have spent some time thinking about it more. And that is, the fourth factor turns out to be sort of a return to the past. But, for example, in this RSA research paper, these guys have put some science to it. The fourth factor is someone you know. So of course that's the pharmacist and Opie example, that is, the pharmacist trusted Opie because he knows him.

Now, in an electronic mode, of course, how do you prove that you're someone you know? That is, the idea being we're still talking about trans-Internet authentication. Well, in the RSA research paper - they're of course a little RSA-centric. We can't fault them for that. So they talk about a model for the fourth factor involving their SecurID token. The way this would work is, imagine that Sally - we'll just make up a Sally - has lost her token, but needs to authenticate herself in her corporate environment. So there's a formal structure to allow someone Sally knows who has not lost their token to extend the authentication through his trust and his knowledge of Sally. So what happens is, Sally calls Bob and says, Bob, I've done it again, I'm sorry, I'm so ditzy this morning, I left my SecurID token at home. I need you to authenticate me.

Leo: Uh-oh. That's the kind of thing a spy would do.

Steve: Well, now, that's a very good point. And this is one of the weaknesses that this RSA security paper - and I will have a link to it in our show notes so people can dig down into this if they're interested. It's one of the weaknesses that these guys not only qualify, but quantify, in terms of how does this weaken the system. Because they talk about, for example, if Sally sent Bob email to say "I left my SecurID at home," that's obviously very weak form of someone you know because email is so easily spoofable. And so they design a system where, in a drop-down list box, in order for Bob to authenticate Sally to his corporate network infrastructure, for example, there's a drop-down list box where there's a number of choices of how Bob knows this is Sally. And it says, "Talked to her," or "Met her face to face," "She called me on the phone," "She sent me email," or, you know, a number of other things. And so the idea being that even things that might be disallowed are listed there so that Bob doesn't lie, so that Bob is encouraged to say, oh, she sent me email. Well, then when he tries to submit this...

Leo: [Buzzer noise].

Steve: Exactly.

Leo: Sorry, Bob.

Steve: Sorry, Bob, that's not okay. Tell Sally she's going to have to do better than that. She has to prove herself. And so what they've tried to do is, because this is obviously about human factors, and anything like this is prone to social engineering attacks, they've tried to give Bob the opportunity to tell the truth about why he believes this is Sally so that he won't just say, oh, it's probably Sally.

And now the other good thing is that - so, okay, just to close this loop. So Bob, believing this is Sally, the way this works is Sally has said to the system first, I don't have my token. Give me a

one-use password that I can use in order to get myself authenticated. So the system gives her some sort of - a token of some sort. She then provides that to Bob and says, Bob, I left my token at home. Could you get me on the network, please? Can you authenticate me?

Bob says, well, this really does sound like you, Sally, so yes. He has some credible reason to believe this is she. So he goes to the system and says, hi, this is Bob, and I've got my SecurID. So he goes through the normal protocol of authenticating himself to the system so that the system knows this is him. And he says, here is the token that you gave Sally a couple seconds ago, which she needs to use to authenticate. So he provides that to the system. The system says, okay, this is what we gave to Sally a minute ago. We know you're Bob. You've proven that to us. So here is the matching, like, magic token that you are to give to Sally to prove that you're who you say you are, and you're known to the system.

So the system gives something back to Bob, this matching token, which he then gives to Sally, and Sally then provides to the system. So what this does is this securely and, in a one-time-only use, it creates a series of chains where the only thing you really need to depend upon is that Bob and Sally are in agreement that they know each other; or, more specifically, that Bob is sure this is Sally who has made the request. And so Sally, in returning this token that she received from Bob, verifies that she's had this dialogue with him. The system verifies that this matches the temporary token it initially gave to her, and the only way that can happen is if she's been the recipient and this thing has flowed through Bob to the system and then back through Bob to her. So, I mean, it sounds confusing. In practice it's not very difficult. And...

Leo: Is this the web of trust that they talk about?

Steve: Well, no, that's - we're going to get to that in a second.

Leo: It's not, okay.

Steve: And I expect you have much more experience with that, Leo, than I do because you have been for many, many years a PGP user.

Leo: Right, right.

Steve: So this is not a web of trust. This is sort of a fallback solution for what happens in - or a solution to the problem of the strength of a two-factor system losing its strongest factor, meaning the something I have, unfortunately I left it home, so how do I authenticate myself in an online fashion, in a secure fashion, in the absence of one of these tokens? And the idea being that, if the players play by the rules - and of course that's the weakness in this is, again, we are talking about the human factor side. But any system like this that is strong needs to have some sort of - well, and to be practical, needs to have some sort of fallback methodology.

The classic example is when we use passwords to authenticate ourselves onsite when we're signing up with our account, they'll often say, well, to give us some personal information about yourself, what was your favorite teacher in high school, your first pet's name, you know, what city were you born in? And of course those questions are used exactly in this fashion as sort of fallback password-recovery options. The problem there is, the weakness of that is those can often be determined through data mining. For example, it's possible to find somebody's mother's maiden name on the Internet through data mining. And unfortunately, you know, mother's maiden name is the classic fallback authentication recovery approach. And it's...

Leo: And everybody knows it.

Steve: It's terminally weak, yeah. I mean, it's really not...

Leo: First of all, it's easy to find somebody's mother's maiden name.

Steve: Right. And the other problem with that is it's inherently static. Everyone has one mother, and she had one maiden name, and it's not changing. The nice thing about the model that the RSA guys came up with as a fallback solution for the temporary loss of a token is it does use one-time-only identifiers, so it's not subject to any kind of a replay attack. That is, it's not possible for someone else to use the token they got from Bob or that Sally wrote down on a pad of paper or something and use it as a single-factor authenticator to the network.

So it's a cool system, and it demonstrates, again, a way that it's possible to use someone you know who is essentially vouching for you. The point I was going to make also is that Bob knows he's on the hook for having authenticated Sally. So given that logs are being made - and logs are always made in these sorts of secure authentication scenarios - there's a log entry that says Sally was authenticated through Bob at this time. So if it turns out that Sally has done something wrong to the system, or it turns out this wasn't Sally, Bob knows he's responsible to the powers that be, basically, of providing the credentials, the temporary credentials that this person who claimed to be Sally was.

Leo: Had.

Steve: Did.

Leo: Some word in there, a verb. We'll just put "your verb here."

Steve: Exactly. Now, the web of trust is sort of a longstanding variation on this. The idea - we've talked about the public key infrastructure, or PKI, in the past. And we've discussed it extensively as regards the signing of security certificates. You know, GRC, my web server, supports a secure sockets connection, an SSL Secure Sockets Layer connection. And we enforce that at any time during, for example, our customers' use of the ecommerce system for purchasing SpinRite. My certificate is signed by VeriSign, that is a trusted authority and that signs many people's certificates. So the idea being that the browser trusts VeriSign to do its due diligence in verifying that anyone applying for a certificate for Gibson Research Corporation really is me, essentially, chief honcho at Gibson Research Corporation. And I do have to go through a bunch of hoops, you know, verifying phone numbers and addresses and Dun & Bradstreet listings and that kind of stuff in order to get my certificate every couple years when I need to renew it. So there the idea is we have a central certificate signing authority, a certificate authority that signs certificates. And so our trust of them is extended through their signing action to the certificate. Well, Paul - it's not Paul Zimmermann. I'm blanking on his name.

Leo: Philip.

Steve: Phil, Phil Zimmermann. I knew it was a P. Yeah, Phil Zimmermann, when he created PGP, he wanted to create a public key infrastructure; but he did not want to tie it to a similar

sort of central signing authority. So his idea was to create a so-called "web of trust," the idea being that you could have people sign each other's public keys, their PGP keys, and thereby in doing so saying I assert that this person is who he is. And then the idea being that, in the ideal case, for example, I may not know - we'll stay with Sally and Bob. Sally may not know Bob. But Sally knows Jeff, and Jeff knows Bob. And as it happens, Jeff has signed Bob's key, his PGP key, and Sally has signed Jeff's. So there is a three-person chain involving two signatures that create this three-node chain. So the idea that, if Sally trusts Fred because she knows him, and Fred is asserting by having signed Bob's PGP key that he knows him, she can be very sure that she has the key that Bob is asserting.

Leo: Right, like Bob and Carol and Ted and Alice.

Steve: Exactly. And...

Leo: Bob knows Ted, and Ted knows Carol, and Carol knows...

Steve: Exactly. Now, the problem of course with this is, as the chain gets longer, again we're still basically in a social engineering mode. That is, as is always the case in any authentication scenario, the trust that you can have in the authentication provided is only as secure as the weakest link. So if you did extend the trust really much further beyond this two-link example, suddenly you're probably trusting somebody you don't know. So you don't really have personal verification. See, in the simple two-link model, Sally does know Fred, and so she knows how trustworthy he is. And she knows he would not assert that he knows Bob unless he really did.

Leo: Unfortunately, we live now in such a fast-paced world that, especially with PGP, you don't know everybody personally, directly.

Steve: Right. Well, and you're certainly not - not only do you not know everybody personally, but you certainly don't - you aren't just so-called two degrees of separation.

Leo: Many, many more.

Steve: Exactly. And in theory, of course, I think studies have been done, and in fact we talked about at some point during our 94 episodes this notion of six degrees of separation. Studies have been done that show that, in most real-world social networks, any two people on the planet are connected by only six links. There's six degrees of separation between any two people in general, you know, on average. So the beautiful concept of this notion of a web of trust would be that, if PGP were pervasive enough, and everybody got keys and signed them and had everybody that they communicate with sign their key, and they signed theirs, that you would end up with a very useful and robust web of trust.

Now, again, trust is the key because every link in that web needs to be a trustworthy link and to be worthy of the parties trusting each other. If that's the case, then you really are asserting through, technically, any distance, any number of links - obviously, if every single link is something you can count on, then even though you might be ten steps away from someone else, if every one of those links is trustworthy, then you can be absolutely sure, to the degree of the least trustworthy link, that ten steps away this is really somebody who is asserting that they are who they are.

Leo: One way they - I think in the early days - of course, I mean, there are some obvious problems with this. For instance, my keys are signed by hundreds of people, probably none of whom really know that that's my key, because they got an email from me; or they saw the key, and they assumed that that was emails from me, and they signed it. And I've done the same for others. But really, in the old days you might actually, in fact I think they still do have key signing parties, you might actually physically meet up.

Steve: I was just going to say that, yes.

Leo: And that solves that problem.

Steve: Yes. Many times you'll have SIGs, you know, Special Interest Groups in computer clubs, and they'll get together, and they'll show each other their driver's license to say, look, this is really me. See? This is my photo, and here's my name. I want you to sign my key. Because essentially the more people who sign your key, the more valuable that is because there is integrity associated with it. So there you don't necessarily have a connected web between points, but you're able to get somebody's key and see how many people have said this is really him. And so you say, okay, what are the chances that all these people are wrong? And so there you...

Leo: Right. It's certainly possible if they're all befuddled and fooled, but...

Steve: Yes. And that's a very good point. It would be certainly possible for some bad guy to go around with a fake ID and use key signing parties to get a ton of people agreeing with him that he's bad - I mean, that he is who he is when in fact he's not, and then take advantage of this robust impersonation to do bad things. It's funny because an example of this, the same sort of social networking is what we see with eBay purchases being verified by a chain of people you've bought things from before. I know that, when I'm buying things on eBay, I always make a point of checking out somebody's reputation. So reputation systems are sort of a softer...

Leo: Same idea, aren't they, yeah.

Steve: Exactly, the same sort of thing.

Leo: And we've seen them fooled before.

Steve: Yes, and in fact there are bot networks now that are designed to create spoofed reputation of bad guys on eBay by...

Leo: They sign each other's certificates, in effect.

Steve: Essentially do that, exactly. And so they create a fake internal web of trust which unfortunately is extended into the normal eBay web because they're able to create a deep and rich-looking background of successful purchases of something, basically a penny each were the purchases, so not much money got spent. And that same penny probably went back and forth within this network of bots.

Leo: Exactly. Well, so you could see the problems inherent in this. It's not exactly a perfect form of authentication. But I guess for PGP and the purpose of PGP it's sufficient.

Steve: Well, yeah. And it is - exactly. It's not like you're going to rely on this absolutely. What you really want is to meet up with the people you care about, like you and I, Leo, you and I get together physically in Vancouver...

Leo: We should sign our keys.

Steve: We sign each other's keys, and game over. There's no way now that anyone could spoof you to me or me to you.

Leo: And it would be sensible at that time to look at a driver's license or some other form of authentication, maybe more than one, to make sure that that's the case.

Steve: I suppose that would be true if you and I...

[Talking simultaneously]

Leo: ...know you.

Steve: Yeah, exactly.

Leo: But we don't even need to meet up because you could say, okay, right now I'm going to send you an email with my key in it, would you sign it. That would work, too.

Steve: That would work, you're right. Because I was just trying to think whether any man-in-the-middle attack could attack that. But no, because I've signed my own key. So you could get my public key and verify that that's my signature on my own key, and nobody else would be able to spoof that.

Leo: Right, right.

Steve: So you're right, you could establish trust in that fashion.

Leo: Cool.

Steve: It is. And I found something that I'm excited about. I've got one on order. And I'm excited about it because it's cool and it's cheap. I found a - it's like a $20 1-gig USB storage device with a built-in fingerprint scanner.

Leo: Oh, clever.

Steve: $20. Anyway, so I need to check it out and see how it works because they talk about how it only really works on Windows 2000 and XP. It's like, oh, okay, well...

Leo: I'll tell you why. We've actually shown that and demonstrated it in a lab. And it's because it's got software on it; right? And so I don't know if it's a U3, same concept as a U3 USB key. But it's the same concept. Whether it's the same software, I don't know. But it automatically launches the authentication software. And it has to run - it just has to run on XP or 2000. There's no reason it couldn't run on Vista. They just apparently haven't written it for Vista yet.

Steve: Okay. What I was hoping was that this thing did this internally, which would have been so cool.

Leo: Well, it does have an internal encryption key, a unique internal encryption key. So, yeah, but software running on the key, I guess not.

Steve: And in fact I intended to spend some time talking about U3. The problem is, it gives me a queasy feeling to stick a USB dongle into the system...

Leo: It should, it should.

Steve: ...and have it launch software, and for there to be no means for me to override that. It's just...

Leo: Right. In fact, we've seen that demonstrated. I've seen hacker attacks that involved putting on a U3 USB key a hack keystroke logging software, rootkitted, so forth. You plus it into a machine. The software checks to see if it's been installed yet. If it hasn't, it installs itself. The hacker walks away. This happens within 30 seconds. Then comes back a week later, same key, checks, says, ah, I've been installed, let me download the database of logged passwords. So this is a very, very dangerous technology. And if you use a machine in a library or an Internet café, I think you should assume it's been compromised in this way.

Steve: It's probably the perfect example of the reason why that's too convenient. I mean, we talk often about the tradeoff between convenience and security. And it's like, yeah, it's really convenient to plug this thing into a computer...

Leo: Really dangerous.

Steve: ...and have it launch. But oh my goodness, is it dangerous.

Leo: Really, really dangerous. Well, you know, I guess I would hope there would be some

way to disable that autorun in Windows. But I'm not counting on it.

Steve: There must be, although in my looking at it, it creates a virtual CD drive and then another data drive.

Leo: Okay, good. Because you can turn off auto-mounting CDs.

Steve: Right. And in that case, so that's the technology that it's using in order to take over. So, yeah.

Leo: Very dangerous.

Steve: Although I think to remove it you need, at least when it was in, I needed to use their removal software to remove it, which of course is the first thing I did is I get this thing - this is a bad idea. Get this off of this dongle. I want all my four gigs back.

Leo: Anyway, that's right. So don't buy a U3 key, and disable it on all systems that you want to keep secure. And by the way, if you're a hacker, you can check out, I think it was Hak.5. And Darren Kitchen told us how to do this, so go to Hak.5, Darren will tell you exactly what you need to know to make a bad, bad thing.

Steve: Well, in the future we will be covering various aspects of authentication. And you know, Leo, I've got to say, the more I think about this SecurID token idea, the more I like it. I mean, I'm wishing, for example, that there were a way to use - that individual end-users could get something like the SecurID token, and that there would be a service offered that websites could use to authenticate people, that is, rather than using my email address and a password, which continually makes me feel uncomfortable because I'm using static information to authenticate myself. It's difficult not to reuse that on other websites, I mean, it takes tremendous discipline not to.

Imagine if instead, like all websites - well, that's not going to happen, but a large number of websites, or maybe some trusted entity provided a service where websites could say, okay, prove you are who you say you are, very much the way that this RSA model follows. They give you a token. You use some little app on your PC - again, obviously you're on the Internet already. You authenticate on the fly with some third-party authentication service that would use something like this kind of a time-varying token. You provide that service with a token the site you want to authenticate to has provided you. The service encrypts it or signs it or mutates it after authenticating you. You then provide - you just cut, copy, paste. You drop it back into the website. So the website knows more securely that you are you than if you had given it your email and password because the website has no idea who's just done that. And you're more secure because you've not had to give it anything that you've used anywhere else. And it's never the same twice. It's one-use authentication. I just...

Leo: I think we're very close to that. In fact, it may be somebody's already doing it, and we just don't know about it. But, you know, the OpenID infrastructure could absolutely support that. And right now most of the OpenID providers I know of don't have a really good, secure authentication system. You know, they just do it by email. But it would be great if somebody came along and did that because there is the infrastructure to use an

OpenID site as an authenticator.

Steve: Right. And so the idea would be, if a user didn't want super-secure authentication, they wouldn't have to purchase a hardware token. But I'd do it in a heartbeat - they're not that expensive - in order to have the ability to log onto sites in a secure way that strongly authenticates me because of something I know and something I have. And even if the site, I mean, the beauty of using a third party is then not every single site has to support all of the infrastructure.

Leo: Yeah, it's very simple, they just support the OpenID infrastructure, and that would do it. There's got to be an OpenID provider that's doing this. And if not, there's a really good business opportunity.

Steve: I think so, in the future, without question.

Leo: I'd do it. Yeah, I would do it. This authentication stuff seems like it's so important, and it almost in some way seems like it's in its infancy. And yet of course there's a lot of work been done.

Steve: Right. The beauty is, as we talked about many, many, many moons ago, crypto is mature enough now, it is in the public domain, we know how to do crypto. And crypto provides so many beautiful solutions. I mean, the idea that you could have a certificate, that somebody can sign it, and that that extends their credentials to your certificate; the idea that you're able to publish a public key, and people can use that to encode and encrypt things that only you, who have the matching private key, can decrypt.

Leo: So elegant, so beautiful.

Steve: It's just, yeah, beautiful building blocks. And I really believe, Leo, that downstream, I mean, I can't think of anything more mission-critical than continuing to move forward on secure authentication technology. I can foresee the day when a huge number of people will have little tokens literally on their key rings and just use them as part of going about their normal day.

Leo: Might be how you get in your car, for crying out loud.

Steve: This is something I have.

Leo: Yeah, I think it's just great. Well, thank you, Steve. A fascinating subject. And there's obviously room to talk more about it, and we will. And we'll talk more about security in general every Thursday on this show.

You can get a copy of the 16KB version, for those of you with dialup connections, at Steve's site, GRC.com. It's also where he puts the transcriptions so you can read along, show notes, and more. That's GRC.com. And also find, of course, ShieldsUP! and all his free security software there, and the great SpinRite, the ultimate disk maintenance and recovery utility. It's a must-have for everyone's toolkit. That's at GRC.com.