SECURITY NOW!

Transcript of Episode #118

Listener Feedback #28

Description: Steve and Leo discuss questions asked by listeners of their 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 they have previously discussed.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-118.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-118-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 118 for November 15, 2007: Your questions, Steve's answers. This show, and the entire TWiT broadcast network, is brought to you by donations from listeners like you. Thanks.

Time for Security Now!, Leo Laporte here. Steve Gibson is in Irvine, and we are ready to talk about protecting yourself online. Hey, Steve.

Steve Gibson: Hey, Leo, great to be back with you again.

Leo: Little disappointed because I had high hopes that we could use this new iChat client that's supposed to use a really high-quality codec. But we're still using Skype because Skype works. And iChat did not. I mean, it worked, but it was awful.

Steve: Yeah, and we really, you know, we actually spend a lot of time dialing Skype in over time. And I think maybe if we had given iChat some more time, I mean, for example, I don't know anything about it relative to relaying and passing through our firewalls and all that. I've got static ports mapped for Skype and so forth, so...

Leo: Yeah, we didn't do any port forwarding. So maybe that would help. But still. You know, AAC-LD, the codec they're using, or purportedly using, is the same one I use for the radio show. It's a very high-quality codec, designed for - LD stands for Load Delay. It's designed for latent networks, but...

Steve: Hey, you know, we didn't set any changes or specify the codec. Is it just changed globally?

Leo: Yeah, maybe - I'll do some research. Because, you know, Amber and I want to use it, for not just audio but also for video. So I'll have to see if we can figure out how to get it working. There's other stuff we might try down the road, too. We're always looking for other ways. Skype's been so good, though, it's hard to beat Skype.

Steve: Yeah.

Leo: Congratulations, by the way, you're now on the Zune.

Steve: On the who, what, what?

Leo: On the Zune. You've heard of that, have you? It's a Microsoft product.

Steve: It's that brown thing, isn't it?

Leo: Yeah, well, now they also have khaki green, army green, and some other colors. The Zune...

Steve: Get a clue.

Leo: Well, it's a little better. The new Zunes actually look like only one-generation-old [indiscernible]. So that's not, you know, they're getting close. But the thing, and I won't be flip about this because I really am excited about it, they are supporting podcasts finally, natively. I mean, there's podcasts right on the front menu. The Zune Marketplace now has a podcast page, makes it easy to subscribe. And if you go, if you are a Zune - own the new Zune, or if you've updated your old Zune, and you go to our Security Now! page at TWiT.tv and check the subscribe links, you can now subscribe via the Zune Marketplace. You just select Zune from your list. So I'm just, you know why I'm excited about it, with a big company like Microsoft behind podcasts - Yahoo! has dropped out, Odeo's dropped out, PodNova's dropped out, iPodX has dropped out...

Steve: Whoa, whoa, they have?

Leo: All of these people are gone.

Steve: Wow.

Leo: And so, frankly, it's an iTunes world. And while I'm thrilled that at least somebody still supports podcasts, it's great to have another player in there, and a big one. Now all they have to do is sell some Zunes.

Steve: Well, that would be good. And of course it's also good just in general from a standpoint of increasing the potential listener base. I mean, you know...

Leo: Well, that's exactly my point. I mean, we're frozen now. You know, we...

Steve: Yeah. When you first mentioned to me, what, two and a half years ago, more than that I guess now, coming up on three years, you said, hey, Steve, how about doing a weekly podcast? I said, a what cast? Literally, I had never heard the term.

Leo: People still say a what cast, and that's the problem. I don't have an iPod, they say. So but, you know, we grew very quickly. You're the number two podcast on the network, right after This Week in Tech. And it's, you know, obviously still very popular. But it hasn't grown much in the last few months, and it concerns me. And I think that that's really that we've just saturated the iTunes listeners. And I think particularly for this and Windows Weekly, which are really more Windows-centric, it will really help to have Microsoft in our ballpark. So anyway, thank you, Microsoft. We also have a Microsoft update.

Steve: Oh, boy.

Leo: A no thank you, Microsoft.

Steve: I was just going to say, yes, and being a security podcast, you know, being tied in with Microsoft, you know, makes some sense here. Yesterday on - or, I'm sorry, day before yesterday, on Tuesday was the standard second Tuesday of the month update. And I wanted just to bring to everyone's attention that this one is really important. It's regarded as not even just critical, but highly critical.

Leo: Oh, boy.

Steve: This is something they've been working on for a couple of months, and it's actually a vulnerability which, refreshingly, does not involve Vista this time.

Leo: So Vista is not impacted. Because I noticed I didn't get the download on my Vista machine.

Steve: Yes, Vista is not impacted. This is actually a glitch in one of the core XP and Server 2003 DLLs, the shell32.dll. And it has the - essentially the effect is that specially crafted URIs, you know, things like news://nntp, telnet, http and so forth, those sorts of references that also include a percent sign are not being parsed correctly. Now, it turns out that there are multiple sort of exploit vectors for this problem. The real problem is in this shell32.dll. However, Firefox users, for example Firefox v2.0.0.5 is vulnerable, and Netscape Navigator, and IRC client, even PDF files. Adobe has pushed out immediately an update to their Acrobat Reader and Acrobat. They're now at v8.1.1 as a consequence because just downloading and opening a specially crafted PDF that contained one of these URIs to take advantage of this could cause a remote code execution on people's machines.

Leo: Wow. So it's not nearly enough for Microsoft to update its shell DLL.

Steve: Well, no, that really is the problem. The problem is in shell32.dll. Microsoft has said that IE7 also has to be installed on the system because apparently the installation of IE7 interacts with the OS and changes the way URIs are processed by other applications.

Leo: Adobe still has to do an update itself, as well.

Steve: Well, I think they're doing it preemptively. They have done it, although I don't think they had to do it if Microsoft had fixed it.

Leo: So just in case you didn't do the Microsoft update, you would do the Adobe update. Although we're not sure, so you probably should do both.

Steve: Well, and it's worth noting that the original release of this was on October 24, which I believe is when Adobe fixed this. So they fixed their particular vector of this well before Microsoft has. And Microsoft is fixing the root problem. So anyway, for people who are still using XP - I'm not yet using XP. But XP and...

Leo: Wait a minute. Let me get this straight. You're not yet using XP. What are you using, Windows 2000 still?

Steve: Yes, yeah. I heard you talking on one of the other podcasts, I think you were talking to Paul, he said, yeah, well, you know, Gibson's still over on Windows 2000, he says. You could hear him scratching his head. And I'm going, okay.

Leo: I love that. "I've not yet gone to XP."

Steve: But I'm close.

Leo: Someday you will, yes.

Steve: Yeah. That's the plan.

Leo: Well, XP's been so heavily patched now it's probably the most secure version of Windows except for - now we find another big hole in it, but...

Steve: Yeah, yeah. So anyway, this is an important one. As long as XP users run Windows Update or click on the little yellow shield if they don't have their system automatically installing updates, this is one you want to get installed very quickly. Oh, and exploits are in the wild. There are Firefox exploits. There are malicious PDFs that are already using this. So this is not just a theoretical problem. This is something everyone absolutely wants to take care of in order to secure their machines.

Leo: Yeah. Interesting that Vista is not bit. Is that just because it had its own shell32.dll, or is it because it's more secure?

Steve: Microsoft hasn't said. My guess is it will have its own shell32.dll, so it probably never had the problem. And again, we do know that Vista is more secure. Have you seen the new Mac commercial, the Mac ad?

Leo: Yeah, I love it. Vista is giving a speech. Ladies and gentlemen, do not go back to XP. Stay with Vista. And Mac says, what's going on? He says, people are downgrading to XP because it's better than Vista. Oh, wow. That's okay, I already have. You know, it's very funny. It's very, very funny.

Steve: They did a good job.

Leo: However, maybe not completely fair. I don't know. I mean, they're picking up on the fact that a lot of people are saying they don't like Vista.

Steve: I don't, and we know a lot of gurus who tried to use Vista and then thought, well, this just isn't worth the hassle, and they went back to XP, so...

Leo: I've been pretty happy with Vista. And I think one thing we have to say is it is more secure; right?

Steve: It absolutely is. Microsoft really has learned a lot of lessons. And to give them credit, they've done - I would argue they've done everything they possibly could without breaking it any more than they have. I mean, we talk about security as being a tradeoff. And there's definitely more they could have done, but it would have been at the expense of backward compatibility. And that's just something they will not do. And, arguably, probably should not do. So, I mean, they're moving forward, I think, as well as they can. And Vista is absolutely a step in the right direction.

Leo: We're going to get to your questions. It's a question-and-answer session. We've got some really good questions and suggestions and conversation. Our listener feedback sessions really have become our most popular. Any other addenda?

Steve: Two things. Someone did an analysis of the numbers being generated by his PayPal VeriSign token, the little - what I call the "football," the oval thing that we first talked about, the so-called Security Key that PayPal was offering for just $5, and which you could also purchase from VeriSign. What he discovered, he was writing down successive numbers, and he noted that the first digit wasn't pseudorandom at all. It was counting, 0, 1, 2, 3, and back around.

Leo: Let me try that. I just got 415892. I'm still doing that. So it should be 5 and something?

Steve: The next one you get will begin with a 5.

Leo: Well, that's not very random.

Steve: And after that with a 6. Well, what it means is...

Leo: Yeah, 5. Wow.

Steve: Yeah. In retrospect it makes sense because this is a time-based token, and it's going to tend to have some clock drift relative to, you know, galactic central. So it makes sense that, if you didn't use it for a long time, the first digit essentially sort of creates a plus or minus five effect, where the system would be saying - I mean the dongle, the token, the fob, whatever you're going to call it is going to be saying, here's number 3...

Leo: Where I am in the sequence, yeah.

Steve: Exactly. And so that allows a - it allows more drift and more lock for their servers. The bad news is, of course, it weakens the whole system. We were saying that there were a million possibilities. Well, there are, but you can guess one tenth of them if you have any sense for where your token is in its 10-cycle phase with the first digit simply counting 0 through 9 and back again. So it really does reduce the effective strength down from what we thought was a million, down to 100,000.

Leo: Yeah, but really, and this reminds me of the Peabody Paradox from last time...

Steve: That yes, it's enough.

Leo: Yeah, exactly. Of course there's the theoretical issues, but then there's the practical issues. And you still have to guess where you are in the sequence of 10. I don't know if that would change much as it is. Now, you know what I'm doing, I'm going to get that card you sent me from VeriSign. I'm wondering if it's doing...

Steve: No, because the credit cards are event-based rather than time-based.

Leo: Oh, that's right, it doesn't generate a new number till you press the button.

Steve: Exactly, and it's not free running. I don't even think you could fit any kind of a timer crystal in something...

Leo: It's too small.

Steve: ...that is that thin. So it really does make sense. It's a different algorithm and a different approach on the credit card form factor than on the little timer-based footballs.

Leo: So we might be, theoretically, anyway, more secure.

Steve: Yeah, I actually think probably would be.

Leo: I'll just try that. I've got a 6 here. You know...

Steve: You're going to be going, oh, wait a minute, here's a 7. Oh...

Leo: No, now it's a 9, it's different, you're right, you're right. That makes sense. It wouldn't work, and so you don't need to synchronize by time.

Steve: Well, I mean, they wouldn't have - they didn't have to do it that way. They could have, for example, searched their expected code space for the code that was presented in order to sort of synchronize. But it would have taken a little more time over on the server end. And again, you know, the fact that you're getting a unique code, one out of a hundred thousand, that's still arguably enough, given that it's only intended to be one of multiple factors and intended to prevent, you know, various sorts of replay attacks. So, yeah, it's still good.

The last thing I want to talk about, and this is something - actually a link from you when this first surfaced must have been the first way I found out about it, but many of our listeners forwarded notices saying would you and Leo talk about this. And this is this very disturbing breach of faith, essentially, that HushMail perpetrated.

Leo: Yeah, and I feel a little responsible because I've been recommending HushMail since day one. And, you know, Phil Zimmermann, the creator of PGP, has worked with HushMail to make sure their algorithms are reliable. And if you read carefully, you'll realize that there is still a safe way to use HushMail.

Steve: Yes. Essentially what happened was - and there's multiple parties involved. As I understand it, there's something involving Canada or something, I'm not really sure if they were the ones or not. But essentially what happened was that the federal government issued them a subpoena saying we need to get the supposedly secure email for the following people. And it...

Leo: And you might say, what? How could they do that? Because HushMail has said over and over again, even we can't decrypt your mail.

Steve: Right. It turns out that that is sort of true. Sort of true. And, I mean, it's a perfect topic for us to discuss here because I've talked about the acronym TNO, Trust No One. And it turns out that the original implementation of HushMail was extremely secure, but a bit of a pain for users to set up and use. That is, it was a client-side encryption using a Java applet that did the encryption.

Leo: So you would download the applet and run it on your computer.

Steve: And you had to install Java, the Java Runtime on your computer. That was sort of one additional step that people didn't want to take. Now, I'm not sure why they couldn't have used JavaScript in order to do this. It would seem to me that they ought to be able to use just JavaScript, in which case you'd be downloading essentially the encryption every time you went to the webpage. But that's not what they were doing. So you were using a Java applet that was locally encrypting so that what the HushMail servers received was truly opaque, and they were unable to decrypt it. Well, what happened was they weakened the system in order to increase convenience.

Leo: Oh, yeah.

Steve: When have we heard that before? So they switched over to a...

Leo: They still use both systems, but they gave you the option of...

Steve: Oh, good point. Yes, yes, yes. They added the option which was much more convenient for people of not needing to use a local Java client, but instead just using an SSL connection where their servers would perform the encryption for you and store it that way and so it would go. The problem is, as soon as they're doing that, as soon as that's over on the server side, well, they know the password used to encrypt your mail. And so the federal government was able to serve them a subpoena saying we want the email of the following companies. And I guess these were - I think they were illegal steroid suppliers or some...

Leo: Steroid investigation, yeah. The interesting thing is they only know the password briefly, as it goes through the system. They aren't logging them. They claim they're not logging them and saving them. So the subpoena said you need to watch as these guys use the system and record the passwords and record the email so that we can get into it. So that's kind of interesting. It wasn't that they were logging it. But if the Java runs on their side, that's the problem. So you can still use HushMail with the Java running on your side, and you would be still secure, as I understand it.

Steve: And it's really a good lesson. I mean, it is exactly, for example, what we were talking about a few weeks ago when we were talking about the Amazon S3 service and Jungle Disk, which we're going to be talking about in the future. I have confirmed that it is possible to set it up, that is, Amazon S3 and Jungle Disk, as fully TNO, that is, fully trust no one. But it is not the default. So you have to know what you're doing and push it further than it would otherwise. In the default case, Amazon does have the key and could be forced, compelled, to respond to a subpoena and turn over all of your files. So again it's, you know, these sort of things are possible. But unfortunately the easy way of using them isn't necessarily secure. And, I mean, we're seeing example after example where, you know, where companies can be compelled to release what information they have. The only secure thing to do is not let them have it.

Leo: Very interesting. And I think scary because people really felt like, oh, HushMail, we should trust them. They probably should have warned people. I think they did, but maybe should have warned them in bigger letters that by doing - letting them do the Java was potentially compromising.

Steve: Yeah. I would argue that, if they were offering a system with that kind of security, then they should absolutely not have allowed it to change in a way that was insecure, that is, people were assuming what HushMail was saying would continue to be true. When they couldn't come up with a way that would prevent them from being able to respond to, for example, this sort of subpoena, then they should have just discontinued it. They should not have offered an insecure alternative because people are going to take it.

Leo: Yeah. That makes sense. Are you ready, Mr. Gibson?

Steve: Let's plow in, Leo.

Leo: Okay. We can do that. Our first question of the day, from Aye Mossum in Redlands, California. He's using Perfect Paper Passwords. He's using them for his work. He says: Thank you so much for your recent talk on PPP. He just installed the PAM module by Tom Fors onto his own servers at his workplace. He says: I'm the IT department. I am the entire IT department. And I feel so much better knowing that I carry in my wallet the keys to get into my server, that no one else can get in without, like, picking my pocket and some seriously nefarious means. He says: Thank you both for such a great podcast. As a result, every week or two I'm implementing some new security measure in my network. And thanks, Steve, for SpinRite. Oh, he uses that, too. He says: It's saved my bacon twice so far. That's nice.

Steve: Well, I wanted to let people know that we are continuing to have additional open source submissions. Also we're in the process of taking Perfect Paper Passwords to version 3, which is a major update to it. There was a lot of discussion after last week's podcast about this issue of the so-called problems with dirty words, that is, you know, there were a lot of people who were saying, you know, I'd like to use this in a commercial setting, but if this thing spit out a word that...

Leo: A four-letter word.

Steve: Yeah, because, I mean, I use those four-letter passcodes. So it could spit out, I mean, basically every four-letter word is possible.

Leo: Right.

Steve: And so, you know, people were saying, eh, you know, it's just - I'm not going to feel comfortable doing that. My boss would fire me if one of these little passcards spit out particular four-letter words. And so it's like, okay, well, that's a reasonable problem. So we discussed it in the newsgroup. Some people were also just saying, you know, I'd rather have a character set that needed no shift key. Somebody else said, you know, I like hex, and you don't have to worry about four-letter words with hex because...

Leo: Or any words, right.

Steve: ...they're all boring. And so I was sort of thinking about that. And so I thought, okay, well, what if we reduced the size of the character set. We had a 64-character character set. And I experimented with reducing it to a funky number, that is to say, 35. Because it turns out that there's nothing says character sets have to be even powers of two. Normally people think in terms of like four bits would be 16 characters for hex. Five bits would be 32 characters. Six bits would be 64. But you could also have strange numbers. Anyway, what we've done is we've extended the whole concept so that essentially now it's a metasystem. It's a one-time password metasystem that allows any user-specified character set and any length of token, that is, any number of characters. And I've got mine running. A couple other people have theirs running in PHP and C so far.

And what this essentially will do is, it allows you to specify how you would like to use it. If you would like a larger alphabet, and you're not worried about ambiguous-looking characters because you trust yourself or your particular users in order to be able to determine the difference between a zero and an alphabetic "O," for example, you could use a larger character set. It would give you substantially greater security. If you would rather have - there was one guy participating in the newsgroups who said, you know, for a one-time password system, this is already overkill. I've got my userID. I've got my passphrase. I just want to use a three-character token. Well, version 3 of PPP will allow you to do that.

Leo: Any length at all.

Steve: Any length at all. Or you might say, I want to use this thing to generate static passwords, sort of like of the kind that I get from GRC's Perfect Passwords page, but I'd like them to be printed out. So there you could use a longer key that is not changing all the time. So you want it to be longer so that it's stronger. So anyway, we're in the process now of revamping all the software. I've got to rewrite all the pages again. I've done it twice so far.

Leo: Maybe you want to wait for version 4?

Steve: And I just want to get it done. So I'll talk about it a little bit more next week. But then this thing, this topic will finally be behind us. But we'll end up with something far cooler than we started out with.

Leo: Good, good. That's really neat. Version 3. There's never a final version in software.

Steve: Well, I think this one, we've just about beat this thing to death. So, yeah, 3 ought to take care of it. The third time's a charm.

Leo: Brian Scallan in London, he's worried about the security of SSL or TLS: You guys have probably heard about researchers in Israel who cracked the PRNG...

Steve: The pseudorandom number generator.

Leo: ...in Windows 2000. As a result, Windows SSL appears to be compromised, which is worrying because of course online banking and secure transactions rely on it. It's suggested the flaw might affect later versions of Windows and even reveal past and future SSL keys. This has shattered my confidence in online security. Have you any advice that can restore it, Steve?

Steve: Well, first of all, he's completely correct. And this is another issue that is...

Leo: I hadn't heard about this.

Steve: ...right now generating a whole lot of concern and fervor within the security community and with a lot of end users like Brian. Here's the story. The guys who implemented the pseudorandom number generator in Windows came up with sort of an ad hoc, very secure-seeming solution. But it does have some problems. And they're problems that could have been avoided but were not. The nature of the problem is - well, there are several natures of the problem. But primarily, in order to determine the next numbers being generated by Windows' pseudorandom number generator, you need to determine the state of the pseudorandom number generator. It turns out that it's up in user space, and it's a per-process pseudorandom number generator. So, for example, if you're running IE, that creates an instance of the pseudorandom number generator in the Internet Explorer's user space.

Leo: Now, this is true of all pseudo number random generators, like when you say PRNG, that if you know where it is, and you know enough about it, you can tell what the next number will be; right?

Steve: Well, yes, except that you could also constantly be mixing in updates. For example...

Leo: It could change the seed periodically.

Steve: Well, actually constantly. For example, in all Intel-based systems you're able to read the number of clock cycles which have occurred ever since the last reset of the chip. So, for example, that could be mixed in on the fly so that you'd really, you'd have a big pool of randomness. But you're doing something that is constantly churning it. Unfortunately, Windows' pool of randomness is static, and it is only re-randomized after it's generated 128K of random output.

Leo: Oh, that's a lot.

Steve: Well, yes. It's turns out that, based on the rate of which, for example, IE pulls random numbers to generate its SSL keys...

Leo: That could be days, months, years.

Steve: Yes. It could be from the time the user powered on their computer until they turn it off. I mean, so in a whole session-long series. Now, the reason this is not really a big problem is that there's no way somebody in a man-in-the-middle attack, which is really what SSL is designed to prevent, gets any benefit from this. That is, it's only by being in the machine which is initiating the SSL connection, that is, a Windows machine, that would allow you to get the whole state of the pseudorandom number generator, which would then allow you to predict - turns out it's both ways. You can predict keys that have been issued before and keys that will be issued in the future. But you have to be in the machine that is on your end of the SSL connection anyway. So if you're in the machine, and you're a malicious trojan, you can just get the data before it's been encrypted.

So anyway, this is a bit of a tempest in a teapot. It is the case that Windows 2000 and apparently later versions of Windows, although these researchers who did this work did not look at XP and Vista, I wouldn't be at all surprised if Microsoft ends up revving their pseudorandom number generator to be continually mixing in new data. It's interesting, there was a comment on the fifth page of this report from the Israeli guys, they talk about how you could simply use AES in a counter mode to generate as good a source of randomness as all the hoops these guys jump through. Well, AES in counter mode is what Perfect Paper Passwords happens to be using because it is good enough. I mean, it's all you need as a really good cipher. And you just stick a counter on one end of it. Although you would obviously need to be doing more in order to get actual entropy because Windows' applications for the randomness is different than ours is with Perfect Paper Passwords because we want the Perfect Paper Passwords to be deterministic and reproducible and so forth.

But the point is, Brian, who's worried about SSL, need not worry because SSL was not ever intended to prevent an attack on either endpoint. It does not do that; it cannot do that. It was intended to prevent an attack from man-in-the-middle attacks, to prevent people sniffing your traffic and being able to decrypt it. And even this weakness in Windows' pseudorandom number generator does nothing to make it possible for someone being a man in the middle to know what that key is or the prior keys or the future keys. The only way you can is by looking, basically getting the whole state, exactly as you said, Leo. Once you have the state of a static pseudorandom number generator like Windows is using, then you're obviously able to algorithmically march it forward and see what it's going to be doing next, just like Windows does.

Leo: So to summarize, this isn't anything to worry about unless somebody has access to your machine. If they do have access to your machine, there are far many more things to worry about than this. And it doesn't compromise SSL's ability to do what it's supposed to do, which is protect you when you're logging into your bank.

Steve: Right. Exactly. It is meant to protect anybody, that is, SSL and TLS are meant to protect you from anybody listening in on your conversation, and they still can't.

Leo: Yeah. Excellent, thank you. No need to worry, Brian. Lil in Long Island has little WiFi gizmos. Lil says: I've noticed more and more MP3 players and handheld videogames are now coming out with the ability to wirelessly sync to your home network. Since they support WEP security, it's obvious they must store the network WEP key in flash memory. Since these also connect wirelessly with other units, is it possible to sniff the information on the player and get the WEP key for my home network? Hmm.

Steve: Well, I wanted to mention a couple things. First of all, we all know from listening to these podcasts that WEP is now really broken. We did a podcast a few months back called Even More Badly Broken WEP or WiFi. Essentially it's so bad now that there are freely available, publicly downloadable WEP-cracking tools that can crack WEP in about a minute. So it's almost easier to crack it than it is to enter the complex password into your own...

Leo: Faster.

Steve: ...yeah, into your own machine. But relative to Lil's specific question, I mean, it brings up a problem. I've only seen one router that solves the problem, and that is that you may have devices, like many people have had TiVos which only support WEP, or the Wii, or the Nintendo DS may only support WEP; whereas you're running your network with good WPA encryption, which is what you really need. There is a Buffalo Tech router which supports both at the same time, which is really a wonderful solution because it allows you to maintain the full security of WPA for, like, the links between your laptop that you use and your router. But when friends come over, or if you have devices which don't support WPA encryption, then you're able to run them in a less secure mode, conscious of the fact that it is less secure and that that dialogue could potentially be cracked, while at the same time not requiring you to downgrade the security of your entire router to the lowest common denominator. And I really hope we're going to be seeing more routers that will allow multiple levels of security encryption at the same time because people really need to keep their machines running, their main connections running very securely with WPA. And it's a shame to force the whole WiFi network down to the weakest security of the device that you want to use.

Leo: Is there a reason why these devices use WEP? Is it maybe a little bit easier to do WEP, so some of the devices are more likely to use it, or...

Steve: It's just - I think it's just - it was probably in the pipeline longer or...

Leo: They're older.

Steve: Exactly. They're older devices; they haven't been updated. Remember that WPA was deliberately designed so as to have a secure mode that was not more computationally costly than WPA. Because WPA still uses RC2, or, I mean, RC4. It just uses it in the right fashion...

Leo: I see.

Steve: ...not in the wrong way.

Leo: So it's no harder to implement.

Steve: Correct.

Leo: Yeah, we demonstrated, I think it was that Buffalo router on the lab. And it's very cool. And other people just use two routers. They have a bridged router that's doing the WEP. I mean, you have to buy another wireless router, but that's another way to do it, of course.

Steve: Yes, absolutely.

Leo: Adrian Jimenez in El Paso, Texas would love to win the lottery: With your Perfect Paper Passwords system, you say there are 16,777,216 possible combinations of four characters for any given passcode on one of the PPP passcards. While this sounds like a high number, let's put this in perspective, shall we? The odds of winning the Texas State Lottery are one in 25 million. Does this mean I have better odds of guessing one of your passcodes than winning the lottery? Yes. But then he goes on to say: Couldn't you increase the number of characters to five and thereby increasing the odds to one in a billion? I know it's a tradeoff. But people win the lottery all the time. Maybe I missed something, or maybe it's just the case that four characters is plenty, and I'm just disheartened by the realization I'll never be a millionaire. This is a very common statistical fallacy.

Steve: Yeah. What's happening of course with the lottery is that, even though the odds are higher, we've got a ton of people all playing at the same time. It would be like simultaneously guessing all of the possible Perfect Paper Passwords. So if you were able to do so, then yes, it is the case that the PPP system, from that one aspect of a multifactor authentication system, is less able to withstand a massive parallel attack than the Texas State Lottery. Although remember that it's one factor of multiple, so you have to have the user's ID, their secret passphrase that does not change and is hopefully strong, and this one-time passcode that does change every time. So it wasn't meant to stand alone.

However, I also like the fact that he talked about increasing it to five characters, which of course the version 3 of our PPP system does seamlessly allow people to do. And I expect that some people may like, just feel more comfortable with longer passcodes. However, again, the idea of a one-time password system is that it's changing every time. So you've got some limited possible number of codes that an attacker could guess with. They're not just guessing that. They've also got a username and a passcode, or a passphrase, that they need to get correct also. And it changes every time. And after five wrong guesses it's easy for the server to lock out that IP and say, look, we don't think you're really an authorized user, go away.

Leo: People, though, I hear this all the time, well, somebody's got to win the lottery. Somebody's going to win it. And that doesn't mean you're more likely to win it at all.

Steve: Well, yeah. And in fact the perfect example, which is counterintuitive, is that if you toss a coin 10 times in the air, and it comes down heads, it's like, wow, I got 10 heads in a row. The sense is that you're owed some tails, that somehow for like the world to be in balance you have to have more tails now. But it's not the case. It's just - it's very unlikely you're going to get 10 heads in a row. But it can happen. But that 11th toss is still 50-50.

Leo: Just to put this in perspective, he said the odds of winning the Texas State Lottery are one in 25 million. The odds of getting struck by lightning are one in 576,000. You're much more likely to get hit by lightning. The odds of getting killed by lighting are one in 2.3 million. You're 10 times more likely to get killed by lightning than to win the Texas State Lottery. And in fact the odds of becoming a saint are one in 20 million, even better than your odds of winning the Texas State Lottery. So I think you're protected pretty well.

Steve: Yeah, I think so.

Leo: And you are, by the way, the odds of winning the California State Lottery are one in 13 million; so come here, you'll win. You'll have a much better chance. James Earl Ford in Apple Valley - is that Minnesota or Montana? Minnesota, I think. MN, right? Just learned of another multifactor. We were talking about multifactor authentication. I just sat in on a webinar for a product named "BioPassword" that may be a candidate for one piece of the multifactor authentication process. The company website, biopassword.com. If you think it has some possibilities, you might want to review it on Security Now!. I've learned so much from your show. Keep up the great work. Also, more than once, SpinRite has saved disks for me that I thought were toast. Great product. Thank you, James Earl Ford. Here's what BioPasswords says on their site. I've never seen this:

"BioPassword offers the only multifactor authentication software that combines a user's login credential," you know, the login and ID and password, "with the behavioral biometric of keystroke dynamics," that is, your unique typing rhythm. I've heard this before, that everybody has a unique typing rhythm. And they use this "to provide a low-cost accurate security solution that is specific to the user, requires no change in user behavior, monitors and authenticates credentials and is immediately deployable across the organization." Or course, you don't have to buy a thumb scanner or an iris scanner. You just probably run some software that watches them type. Have you heard about this kind of stuff? What do you think of that?

Steve: Well, I thought that was an interesting issue. To me, it seems sort of flaky. I mean, I guess if you had a large enough sample, I mean, I'm sure it's the case that if you had a large enough sample of people typing, you could differentiate people. The good news is, with a userID and password, this is a third factor. So you already know who the person is claiming to be. It's sort of like we were talking about, for example, when I go to check in on our equipment at Level 3, I give them my card, my RFID card to sniff, and they measure my hand. So my hand is confirmation that nobody else has my card, instead of being able to recognize me uniquely. As I understand it, Stanford did a bunch of this study and apparently obtained some patents which this company purchased the licenses to from Stanford and have commercialized this.

What would really be nice would be if keyboards had essentially strain gauges in the keys, that is, so you could measure not only the timing, but the speed and force and pressure. Because, you know, my sense is that in order for this to be really robust, you need more data than just somebody doing a hunt and peck on the keyboard. I mean, touch-typists are going to have, I would think, much more specific characteristics than somebody who's pecking out their username and password, like using one or two fingers on a keyboard. So, I mean, I'm [audio gap] see some real evidence that this provides enough specificity to be something that you can trust reliably. And you do have to have something running on the client side. That is to say, a web server running remotely is not able to get the timing of you entering things on your keyboard. So you'd have to have some JavaScript at the least, or an ActiveX control or something, that's able to be local on the client side in order to watch you typing every single key and measure the timing of that.

Leo: Oh, well. Seemed like a good idea.

Steve: Yeah.

Leo: I'd heard that everybody is unique in that respect. But...

Steve: Yeah, I think that's probably the case. But I just, I don't know, I'd rather have something stronger for a third token.

Leo: And I think you're right. If you're not hearing - you're only timing. That isn't quite as - of course, if it narrows it down to one in a hundred, that's a pretty good third, I mean, third token doesn't have to be perfect. Even if it narrows it down to you have one chance of a hundred being the same. Right?

Steve: No. In fact, and that was my point about saying that, for example, with Level 3 I have my passcard which says, okay, this is the one we issued to Steve. And then it measures my hand to see if it believes it's really me. So it's not that my hand uniquely identifies me. My hand simply confirms.

Leo: Confirms, confirms, right, right.

Steve: Yes. And so...

Leo: And so probably there aren't that many different hand measurements.

Steve: Exactly. I'm sure other people have the same hand size that I do. And it would be very insecure to try to use my hand to just open the door all by itself, yes.

Leo: Michael Peksa in High Wycombe, U.K., wants some free security consulting: Hi, Steve and Leo. A question, if I may. I've caught up with the entire back catalog of Security Now! over the past four months. It's about eight days of listening. Oh, wow. I'm trying to think of the secure implications wherever possible. My company is rolling out a web application to a large client. The client wants to make access for his users as easy as possible. I want to make it as secure as possible. We're going to restrict access to the web server to only allow our IP range and the IP address range of the client through. The application itself is SSL-encrypted throughout.

The client would like to remove passwords to access the app. Since we'll all be sure that a connection comes from the premises, because they'll look at the IP addresses, how safe is this? I've raised the issue that passwords, however imperfect, will help prevent casual, unauthorized access, say by a visitor to their HQ or by a cleaner late at night. If someone malicious were to get access to their corporate network, that person would have access to much higher value confidential information than he'd get on our servers, so I'm not overly concerned by this. But I have a gut feeling that I'm missing something obvious - oh, yeah - and I'm keen not to leave a back door open to the bad guys. Any advice on this?

Steve: Well, I thought this was an interesting question because he's asking, essentially, he knows that this feels risky, but he's wondering if...

Leo: Why.

Steve: ...basically if IP address range restriction is sufficient.

Leo: So let me understand this. He's saying we know our IP address. By looking at the incoming IP address we'll know if somebody is doing this from our network. If they are, we say go ahead. If not, we don't let them in.

Steve: Well, yeah. The idea would be they're going to be offering some sort of web-based services which they want to - for which they want restricted access.

Leo: This is how Intranets have worked for years.

Steve: Yes. And, well, but it's going to be going out across over the Internet, and it'll be SSL encrypted. The client, who's a remote corporation, the client says, hey, we've got a block of IP addresses that are not changing. They've been assigned by our ISP. And we want to make access to these web services as easy as possible, meaning we don't want our users, that is, our employees, to be hassled about passwords. So set things up so that anyone coming from our range of IPs that are static and assigned by our ISP can have access to the web systems, to these web services.

Leo: Well, I could see one problem right off the bat. It's very easy to spoof IP addresses.

Steve: Well, but remember, not for TCP connections.

Leo: Oh, it's an SSL connection.

Steve: Right. And so SSL runs over TCP. So there is no way to spoof it. And so I don't really think Michael has missed anything. I think he's asking for some excuse he could use to tell this client that this is not very safe.

Leo: Right.

Steve: And frankly, I mean, as long as he understands, and the client understands, that any connection coming from the client's IP range will be authenticated, then - and there are many ways for that to happen. I mean, Michael mentions the night janitor could have access. So that does raise one possibility, and that would be to say, okay, restrict access to working hours during the weekday. So from 8:00 to 5:00 Monday through Friday, if the firewall has the ability to do time and date-based rule sets, then that would be one thing he could do which would lock out the night janitor because then no one after working hours would have access into the web server. Also it's the case that any sort of proxying, any malware that got on any employees' machines, I mean, essentially it seems creepy not to have passwords. But all you're really doing with a password is saying I'm a human who has one additional factor of authentication. I know the password. Maybe you don't need to identify individual people. Maybe they've got cookies on the client machines to identify which machine it is, or the machine's IP. But essentially there really isn't anything that he's missing, as long as everyone understands that IPs cannot be spoofed, but essentially he's statically authenticating any connections coming from that IP range and giving them unrestricted access into his server. What he's written evidences that he understands that. And so with the convenience comes the risk.

Leo: So you're saying it is safe.

Steve: Yeah, I don't see anything wrong with it as long as they understand that it's going to be easy for anyone to access those services from within that client company. But I don't see any other problem with it.

Leo: See, I thought about the spoofing, but you say SSL you can't spoof, so it's safe. They'd absolutely have to be coming from that company.

Steve: Yeah. The one thing you could do also would be to require client-side certificates. That's easy to establish.

Leo: Then you can limit it to certain machines, too.

Steve: Right. And you'd have client-side certificates, you would require client-side certificates for access to the web server. And that would also create an audit trail so that anyone - and so that you knew who, definitively who is using those web services. If the corporation, for example, had their IPs behind a big NAT, then you would be losing that information from clients running through NAT because their local client machine IP would be changed to a public IP when it went across the Internet to go to the web services. But using client-side certificates would give you some additional authentication, and it would allow people, for example, maybe you don't want anybody in the shipping department to have access to this because it's nothing to do with their job. You only want the accounting people to have access to this. So you install those client-side certificates only on those accounting machines. So that's one simple thing you could do to increase the security, limit the access within that block of IPs, and also obtain some accountability.

Leo: And who doesn't like the idea of just having it run without having to enter a password.

Steve: Yeah. It certainly is convenient.

Leo: That's neat. Okay, well, I thought it was a terrible idea, but I was wrong. Bill, with nicely trimmed grass in Grand Rapids, Michigan - I'm sure we'll learn why we say that. Steve does these, it's so funny - writes: I've been an avid listener of Security Now! since Episode Numero Uno. I listen to it while mowing my lawn - ah, see, the explanation becomes clear - which has made that tedious chore much more enjoyable. I actually look forward to getting the lawnmower out now. My question is regarding passwords. I've been using a long random password from the GRC website for a year now. GRC.com/passwords. Of course I can't remember it, and I don't want to type it in each time, so I saved it in a simple text file in My Documents. I worry that perhaps the bad guys might find it and guess what it's for, so when I copy and paste it into the password field, I replace a few of the characters with my normal, not-so-random password which contains letters and numbers. Does this decrease the effectiveness of my totally random password, or have I come up with a useful idea?

Steve: That's a great idea, Bill.

Leo: Makes it unique.

Steve: It absolutely does. All the other debris in the super-long password, it just makes it impossible to brute force it. But if you say, okay, an attacker could take that existing password and try to find some small changes to it that would crack it, but it's just - it's so unlikely. He talks about how he adds his own password that contains letters and numbers. I mean, there's just no way anyone is going to figure out what this guy has done. So I think that's a great idea of essentially statically having a bunch of random junk to which you add your own password, so that by itself it's not useful, but together with something you know and something you have, meaning that little file, you've got multifactor authentication.

Leo: So moving along, Andrew in Australia had an interesting question about MAC addresses: Is there any way, he says, for a server or website to receive and block a MAC address of the hardware connected to it? He asks this because the user group he contributes to is running onto problems with people who are consistently registering accounts using proxy servers after their regular IP address is banned. You see this a lot in chatrooms, too. I know that MAC addresses can be changed, although this would stop a large majority of troublemakers who are not overly computer savvy, in other words, they know how to use a proxy server but not change their MAC address. I've tried by blacklisting known proxies, although this also seems to affect legit AOL users. Can you please help me?

Steve: Well, the problem with using a MAC address is that it never leaves the local LAN.

Leo: You can't get the information.

Steve: Exactly. The idea is, MAC addresses are used to transport the IP packets from one machine to another within the local area network. But when it crosses a router, what a router does by its nature is it takes the MAC address wrapper, actually the Ethernet wrapper because it's an Ethernet packet while it moves over the LAN. It takes the Ethernet envelope off, leaving an IP packet. It then decides, based on the IP address, which of the various router interfaces that packet should be sent to. It then - the packet moves to that interface, and it gets wrapped with the Ethernet packet for its transit over the next link of the router. So essentially, if you look at the Internet as being a whole bunch of routers that are linked together, and each of those links is a little LAN, the packet is constantly being changed. The Ethernet wrapper gets taken off, it goes to the next interface, a new Ethernet wrapper gets put on, and that allows it to transit to the next link of the network. So a packet essentially has as many different MAC address and Ethernet wrappers briefly encapsulating it as there are jumps from one machine or one LAN to another as it crosses the Internet. So when the packet finally comes to you, the MAC address, that is, the source MAC address will be of your own router. Your own NAT router will be the device which most recently sent that packet over your own LAN to you. So unfortunately there's no way to know what the original MAC address of the actual sender was.

Leo: Oh, well. Is there anything he can do?

Steve: I can't think of anything. I mean, if they're creating accounts, I mean, you could do things like we talked about before, for example, like client certificates, and revoke the...

Leo: That's complicated, though.

Steve: Well, it's complex, and it really doesn't solve the problem because bad guys are just as capable of masquerading as a new person, and then they misbehave themselves, you revoke their certificate, and they masquerade as another new person. In an open environment, it really is a problem. And it's just part of the - it's the fundamental abuse of the freedom and flexibility of the Internet that unfortunately it's a mixed blessing.

Leo: It's the bane of anybody who's ever run a chatroom or a forum or website of any kind where people log in. You should see the amount of comment spam I get on my blog. I mean, hundreds and hundreds of them a day. And it's just, you know, there's nothing you can do about it. You just have to filter it out.

Allan Blomquist in Belmont, California is absolutely right. Allan says: I was just listening to your discussion of true one-time use versus pseudorandom passwords in the latest Security Now!. People love this PPP thing.

Steve: It's been incredibly popular, Leo. I mean, it just captured everyone's imagination, I think.

Leo: Yeah, it's fun. He says: I do have to disagree with the conclusion you came to at the end. You said that the existence of keystroke loggers will tend to bias attacks towards replaying recently used passwords. But this is certainly not the case for an attacker with perfect knowledge. Unless you're relying on a form of security through obscurity, the attacker is going to know what your policy is regarding password generation and would therefore choose to cross off recently used passwords, in this case, as opposed to replaying them. Granted, this would increase their odds of correctly guessing the next password by a negligible amount. But it is an increase nonetheless, not a decrease as intended. You'd have to listen to this episode to understand what he's talking about. Seems to me that in the face of an attacker with the perfect information, not only about the user's actions, but also about the algorithms and policies on the server side, as well, you really can't beat a completely pseudorandom password.

Steve: And as I said at the beginning, he is absolutely right.

Leo: Right you are, sir.

Steve: And the reason I put this in here is that we got a phenomenal number of virtually identical comments. And I wanted to acknowledge everyone who said this in various ways. They're right. And the point I was making was we don't know who our attacker will be. We don't know that our attacker will have perfect knowledge. It might be that it's a dumb attacker using a keystroke logger who doesn't know any better than to simply repeat what was recently done. In the case, for example, of making note, as Allan suggested, of the last 20 or 30 passphrases, passcodes, well, so you're subtracting those from, in the first and the second versions of the PPP system, you're subtracting 20 or 30 from 16 million. So, yes, don't guess those 20 or 30 if you know that the system won't have local repeated passcodes. On the flipside, an attacker could be a dumb keystroke logger, and all we were trying to do was just prevent, guard against the case that it was not an attacker who had any knowledge of the Perfect Paper Password system, but just someone who was saying, hey, I just saw this userID, this passphrase, and this other strange code typed in. I'm going to try them again. It's easy not to have those recently used codes present, just to give a little additional defense against an attack that we know exists. We don't know that there's an attacker with perfect knowledge of everything we've done before that exists. But we do know that keystroke loggers exist. So it's like, okay...

Leo: Much more likely.

Steve: Exactly. And so I agree with everybody. Just I want to make sure everyone gets it, that I absolutely agree that pseudorandom is the best, except that given that keystroke loggers do exist, they are relatively common, why not choose a key? And this is what I ended up doing. I did a bunch of research after last week's podcast. And the new code for me has a key finder. You're able to say, give me a good key, a good key being one that tends to have no repeats within a certain window. And it turns out it's easy to find one that doesn't repeat any codes within a thousand. So we get the best of both worlds. We're still using completely random passcodes. It is the case that we're not going to reuse any in a short period. So, yes, somebody with perfect knowledge who knew the system would be able to reduce their possible guesses from 16,777,000 to 16,776,000. It's like, knock yourself out.

Leo: Fun. John in Fort Worth, Texas, has a computer that's a little too memorable. He says: Apparently, banks are now required by the FFIEC regulation - that's the Federal Financial Institutions Examination Council, they regulate...

Steve: Wow.

Leo: ...all this stuff. I looked it up.

Steve: I was very impressed.

Leo: I looked it up - to require multifactor authentication for online banking. My bank has a feature when you first log in that allows it to remember your computer - I think mine does, too, it uses cookies, I believe - effectively bypassing the second factor of authentication, a challenge question, and only use the first factor, userID/password. How does the bank remember my computer? I thought maybe it was using cookies, but I deleted all my cookies and cleared my browser cache. Oh, well, maybe it isn't cookies. I then tried logging in again, and it still remembers my computer. Surely it's not using an IP address as IP addresses can change with DNS. Any ideas?

Steve: Well, my first thought as I was reading John's question was hey, you know, obviously it's just putting a cookie on your computer.

Leo: Yeah, I thought that. My bank does the same thing. If I log into my bank with a computer I've not used before, not location, so if I'm on the road and I have my laptop, it still knows it's me.

Steve: Right.

Leo: It's the computer.

Steve: Right. Well, the answer is, if we're to assume that John's right, and deleted all of his cookies, and it still remembered his machine, there is nothing to prevent something that the client-side scripting is doing from saving local information. That is, maybe there's an ActiveX control. If you run an ActiveX control, all bets are off. You're running basically a DLL provided dynamically by the remote server on your machine. So it could be storing footprints and things in the registry, or squirreled away in some random directory on your machine. So as soon as you're allowing scripting, again, all bets are off. It could be doing anything it wants to to maintain some state over on the client. Which if John's right, that's what it sounds like is going on somehow.

Leo: So even if you delete cookies, it doesn't matter because you're not deleting that particular state-saving exercises.

Steve: Exactly. It's been squirreled away somewhere else.

Leo: But JavaScript can't do that, can it?

Steve: As I understand it, DHTML and AJAX allows you - and AJAX is, you know, the AJ stands for Asynchronous JavaScript. And as I understand, there is some ability to store state over on the client side.

Leo: Tell you what. I'm going to download a completely different browser, which presumably doesn't share cookies with anything, and try to log into my bank. Because my bank knows this machine. So that would be one way to see if it's a cookie-based thing; right?

Steve: Yeah, um...

Leo: No, it wouldn't be, would it, because if the script runs, and it saves it somewhere else...

Steve: Right.

Leo: But if it is cookies, it wouldn't work. But if it's some other system it will work.

Steve: Correct. Although, well, it's hard to say. I mean, there are just so many variables, Leo. And anything could be going on over on the client side based on what code the bank is running on the user's machine.

Leo: Wow, very interesting.

Steve: However, if scripting were disabled, it's difficult to see how that creates an opportunity for any local state to get saved, if it's not cookies.

Leo: Huh. Now I'm puzzled. I just always assumed it was cookies. Well, before this show is over, I'm downloading a browser I've never used before. We'll see. Dan Gardner in San Antonio, Texas wants to keep an eye on his network. He says: Is there any low-cost or free software that will let me monitor network traffic on my own network? I specifically want to be able to track bandwidth being used by each system connected to my router. I get this a lot on the radio show. People want to watch their roommates and make sure they're not using more than their share, things like that.

Steve: Well, it takes a couple things. First of all, the most popular and really spectacular software, which has been developed within the open source community, is now called Wireshark.

Leo: Oh, yes. That was Ethereal.

Steve: That was, exactly, that was Ethereal, which has been around for years and been maturing steadily. It's got a ton of features. And you are able to do things. It's got a statistics package that allows you to sample a bunch of bandwidth and then run statistics over it to, like, get a sense for, you know, in all kinds of different ways. And there are various byte count summation things. So I would absolutely point Dan at Wireshark, which is the new name for Ethereal. And I think it's up, like, at 0.99.6 or something. I just updated my copy actually yesterday. And so it's almost at 1.0. But it's just a spectacular program.

The second thing you need to do, though, is you need to recognize that any kind of a switch as opposed to a hub will insulate or isolate your computer from all the others. So you do need to play around a little bit with your network topology in order to be able to sniff across a switch. What that essentially means is that, as we've talked about before early, early on in Security Now! episodes, a switch provides isolation among its various spokes, essentially. So that if there were other people or other machines plugged into a router's switch, you would be - Ethereal, now called Wireshark, would be able to see your own machine's traffic, but it would not able to see theirs.

In order to see all the traffic in the network, you would either need to plug yourself in on the WAN side of the NAT router, which gets tricky for a whole bunch of other reasons; but more easily would be to, if you had like an older standard Ethernet switch and a hub, then you'd basically, from a wiring standpoint, you would run from your regular border NAT router, you would run that through a hub to a switch, where you plug everybody else except you, and you plug your computer into the hub. Now what's happened is essentially, because a hub does not provide this isolation, you're able to see all the traffic going between these two switches, between the NAT router's switch and the switch which has everything else plugged into it. And then you're able to monitor all the traffic on your LAN. And, you know, you'll find all kinds of interesting things.

Leo: Yeah. Cool. And then you could run a proxy server like Squid on a Linux box if you wanted to gate people's bandwidth. You could say only this much bandwidth can be used. There's all sorts of tricks for doing stuff like that. I turned off scripting, by the way. And I still - my bank still knew who I was.

Steve: And did you delete cookies?

Leo: I haven't deleted cookies yet because I don't want to do that. So I'm going to use a different browser.

Steve: Right.

Leo: Kokaly in Hamilton, Ontario - I'll have a report on that in a moment. Kokaly in Hamilton, Ontario is feeling insecure about Gmail: Is Google's Gmail safe? I'm asking this because when I log onto Gmail, I log in over a secure connection. But after logging in, no secure connection is maintained. The lock icon is gone from my browser, and displaying the page info in Firefox shows the connection is not encrypted. Why? Why? Am I missing something, or what? Are my emails being sent in the clear text? What about when I use a mail client? Are my emails being sent as clear text?

Steve: Well, we've addressed this specifically a couple times before. But it does come up from time to time, so I just wanted to take a minute to remind people about how this works with Gmail. First of all, unless you see that you have maintained an SSL connection, that is, it says https:// while you're looking at your Gmail, while you're opening your notes and so forth, then yes, you are not encrypted. You are not safe. Your email is moving in the clear. What Gmail does is, for example, when you initially connect to Gmail, you'll use something like http://mail.google.com or a number of the various aliases for that URL. The point is that, if you start with a non-secure URL, Gmail will briefly move you to a secure connection only for logging on, and will then revert you to an insecure connection. If, however, you initially connect securely with https://mail.google.com, then you stay secure throughout your entire use of Google Mail. So what you absolutely should do is update your bookmark, your browser tab or whatever it is you've got for going to Gmail, and just add an "S" after the http so that you would initially make a secure connection. And then Gmail will leave you secure throughout the entire session.

Leo: By the way, there's a great - if you're using Firefox, there's a great extension called "Customize Google" that allows you to set that to be always the case, that it's always https, so you don't even have to worry about the bookmark. It just always uses https, which is one of the very good reasons to use it. All right. I deleted cookies on this new browser, and it doesn't know who I am anymore.

Steve: Yeah. I really suspect it is cookies.

Leo: Yeah. And I've turned off scripting. So, yeah, it doesn't know who I am, says who are you? And now it's asking me - it says unable - yeah, it's cookies. It's cookies. So he probably didn't delete them all properly or whatever. But at least on my system here it was cookies.

Steve: Cool.

Leo: Well, Steve, we've completed 12 fascinating questions, 12 fascinating answers. I don't know how you do it every time. But thank you, it's really fascinating. I love the Q&A sessions.

Steve: Well, and our listeners love it. And again, in terms of how I do it, it's just being driven by the questions that we get. We've got lots of smart people who are listening and asking great questions.

Leo: That's why I do talk radio because you know you're talking about what people want to know about. Works really well. Steve Gibson is at GRC.com. That's a great place to go if you want to get Security Now! podcasts in the full version, the full-quality version, or the 16KB version. He's got transcripts, he's got show notes, he's got links to all his good stuff including the PPP page, his security passwords at GRC.com/passwords, and tons of free software including the world-famous ShieldsUP! firewall tester. It's all at GRC.com. And while you're there you might want to take a look at SpinRite, which is without a doubt, as many will tell you, the world's finest hard drive maintenance and recovery solution. GRC.com. You know what we're going to talk about next week yet, Steve?

Steve: We are, yes, we are finally going to talk about this relationship which is disturbing between PayPal and DoubleClick.

Leo: Oh, you've done some research, eh?

Steve: As I said, it is very disturbing. Many, many people have said, hey, whatever happened to that, did it fall through the cracks, did it fall through the cracks? Well, no, actually the Perfect Paper Passwords thing happened, and it continued to burn up more time that I expected. But yes, we're going to talk about what it means that you actually get a DoubleClick URL when you try to download the, well, actually from many links over on the PayPal side. We're going to explain the consequences of that.

Leo: I will be listening with great interest, since I use PayPal all the time.

Steve: I do, too.

Leo: Yeah. Hey, Steve, thanks so much. We'll talk again next week. Is it Thanksgiving next week already?

Steve: Yeah, next week is Thanksgiving week.

Leo: Oh, my goodness. Are we going to do a show on Thanksgiving?

Steve: Absolutely, Leo. We're never missing one.

Leo: So after the turkey, make sure you tune in Security Now! Thanksgiving Edition. Thanks to everybody. We give you our thanks for listening and for all the donations which keep this show afloat and the support you've given us. We greatly appreciate all the participation. I'm Leo Laporte with Steve Gibson. Thanks for joining us. We'll see you next time, Steve, on Security Now!.