LastPass

Description: Steve and Leo cover the week's Internet-related security news, then Steve delivers his long-awaited, in-depth review and evaluation of LastPass. Steve explains the nature of the need for high-security passwords, the problem that need creates, and the way the design of LastPass completely and in every way securely answers that need.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-256.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-256-lq.mp3

Leo Laporte: This is Security Now! with Steve Gibson, Episode 256 recorded July 9, 2010: LastPass Security.

It's time for Security Now!, the show that covers all your security and privacy needs, online and off. Here he is, the man who makes this show, the guy behind Security Now!, the head at the Gibson Research Corporation, GRC.com, creator of SpinRite, the world's best hard drive maintenance utility. He's also a security guru who discovered and coined the term "spyware," wrote the first antispyware, has written many great security programs like ShieldsUP! that he gives away for free on GRC.com: Mr. Steven Gibson. Hey, Steve.

Steve Gibson: And on top of all that, I have to say that I've invested more of my time in the research for this podcast than I have for a very long time.

Leo: All for nothing at all, just because he loves you, the people. Well, thank you. This is actually a topic near and dear to my heart.

Steve: Yes. I know that you're a LastPass user. We've got a huge body of listeners, from the looks of the feedback that I've received, because I've referred to LastPass. I've said, yeah, I want to take a look at it. In fact, I sent a ping off to those guys through email, wanting to establish a contact at LastPass, saying you guys are on my radar, and I'm going to need to get around to it. And so I think it's fitting Episode 256, which of course is a number near and dear to my binary assembly language programmer's heart, that being exactly 2^8, which is the number of combinations, 256 combinations that eight bits can take. So this is going to be a great episode. I've spent at least several days, and now even more than that, using LastPass. So this is a complete security analysis and feature walkthrough on what I have to say is, I think, the best solution possible.

Leo: Ooh. Okay. Well, don't...

Steve: I mean, it's really good.

Leo: Don't give us the lead before we get the details. But I know people want the meat of the matter. And we do apologize. I know this show is coming out a day late. We had a - we normally record this Wednesdays at 2:00 p.m. Eastern, 11:00 a.m. Pacific on live.twit.tv. And if you tuned in at that time you know we were having some sort of connectivity issue. We rewired the entire studio over the weekend, and something went wrong.

Steve: No small job in itself.

Leo: Oh, poor Ken Sheppardson. I thank Ken. Ken is Colleen's replacement, and he's just been dynamite as our chief of engineering. And our studio manager, John Slanina, and our tech wizard Burke, all have worked really hard. Also Kelly from, what is it, it's one light truck, one truck - All-In-One-Truck.com, there it is, thank you, Ken. Kelly from All-In-One-Truck.com who did the - put in the new grid and put in the new lighting and just really did a spectacular job. And of course Alex Lindsay, who recommended Kelly and came in and helped, and Ardell from the Pixel Corps helped. So they were in here. We had five people working their butts off over the weekend. They got it all back together. And we were able to do shows, but we were getting this weird hesitation on Skype. And we're still trying to track it down. We think it might have something to do with the buffering that we also get on our live stream. And so we will let you know.

Steve: We may be a day late, but we will not be a dollar short.

Leo: Good. So coming up, LastPass. Before we do that, though, do you have any updates you'd like to pass along?

Steve: Got all kinds of interesting stuff. I did want to mention that apparently I missed an out-of-cycle Microsoft Windows 64-bit security update, which somehow slipped under my radar last week. I didn't even have a chance or really that much concern, frankly, to verify it. But someone through Twitter said, hey there, you know, I guess he's a 64-bit Windows user, and he said, hey, there was an out-of-cycle update. It's out of cycle because, if that had occurred last week or the week before, well, that was not when we're expecting it. Normally this, of course, as we all know, occurs on the second Tuesday of the month, which is next Tuesday, and we do have some good things happening then. Also Google's Chrome browser, prior to the current version, the current version being 5.0.375.86, has multiple security vulnerabilities. These all sound like star dates now, don't they.

Leo: Doesn't it.

Steve: Stardate 5.0.375.86. Google isn't into big disclosures of their vulnerabilities the way Microsoft has sort of been beaten into, and even the way the Mozilla folks are very open about the things they're fixing. So we don't know exactly what it is that was fixed in Chrome, Google's browser. But we know that there were some memory corruption problems that they fixed, and that's always a little bit of a concern because that's the way that remote code execution vulnerabilities can happen. So anyone using Chrome, and there's more people using Chrome now than there were three or four months ago.

Leo: Than ever before, yeah. It's taking off.

Steve: Yeah, Chrome's market share is growing at the expense of IE and Firefox, believe it or not. So, in fact, I think I said last week that Chrome had exceeded Safari in market share, which is surprising. Wait, maybe it's Safari on Windows.

Leo: Yeah, probably is.

Steve: Yeah.

Leo: Yeah, well, I don't know. That's a good question. We noticed that, as well, and it wasn't clear. You know, Chrome does an automatic update. So for most people, unless they've somehow disabled that, you won't have to think about it; right?

Steve: Good, good. Yes, that is the case. And the Google folks who manage Chrome have said that they are liking Firefox's automatic blocking of insecure plug-ins. Remember that that was a feature that we got, that the Mozilla folks added to Firefox some while ago, a few months ago, where Firefox is now taking some responsibility for noting when plug-ins are obsolete because of security problems and will preemptively protect its users from using known insecure plug-ins. Well, much as Firefox 4 is borrowing heavily from the look of Google's Chrome, that is to say, the chrome of Chrome, the Chrome guys are going to be borrowing that functionality from Firefox. So that's going to be - it's not happening now. But they have said they like it, and they're going to be adding that.

Leo: Yeah. You know, Firefox does notify you that - it downloads it and then says, I've got an update, you want to install it?

Steve: Right.

Leo: Chrome's just silent. You wouldn't know you had a new - I don't think, unless I'm setting it weird, I don't think you'd have a new - know you had a new Chrome unless you looked.

Steve: We discussed last week, as we pretty much always do, Adobe's recent actions and the fact that they did their major update to fix the final problems with the Flash plug-in. And also remember that they fixed Flash first, but Reader and Acrobat were

coming later. And then last week they introduced 9.3.3, which caught those up, that is, their own version of the Flash they were bringing along. Well, we've talked also about this very controversial launching capability which the hackers have figured out how to use. It was Didier Stevens, our listeners will remember, many months ago who came up with a way of sort of finessing the dialogue, the warning dialogue which would pop up to let you know that your PDF wanted to run something, as a way of hiding the fact that what it wanted to run was malware.

Well, with last week's fix, Adobe, thank goodness by now, they are now disabling that launch functionality by default, and they've also created a blacklist which prevents them from running, for example, cmd.exe, the command exe. The problem is, they didn't do it right. And it was only out for a couple days before a hacker figured out that if you put cmd.exe in double quotes, it would run it again. So Didier has on his blog, it's blog.didierstevens.com, he shows a workaround for putting a closing double quote on cmd.exe for anyone who's worried about it. My feeling is, now that they've disabled the launch functionality by default, and they're taking clear proactive steps to close this, this is probably a non-issue. And all of our listeners will have manually disabled that by now. So I did want to say, though, that Adobe's done the right thing, and thank goodness. I'm glad for that.

Leo: Yeah.

Steve: We talked last week about the Windows Help Center zero-day flaw, which is being exploited with increasing frequency. It's really scaling up rapidly now. I wanted to - and this is that it was controversial, you remember, that the Google engineer, the security engineer at Google, Tavis Ormandy, was - he released it after, like, notifying Microsoft on the weekend, on a Saturday; and then he let the news go four days later, sort of surprising them and everyone else, that there was this way to exploit Windows in a way that was, well, that would allow code to be executed remotely, which is the worst kind of exploit. Microsoft has reported that they've detected more than 10,000 PCs infected through this exploit in the U.S. - predominantly in the U.S., Russia, Portugal, Germany, and Brazil, though the highest percentage is in Portugal and Russia, for whatever reason.

The good news is, next Tuesday this gets fixed. So this has been - they're responding pretty quickly. Essentially, remember that this news hit just after their prior second Tuesday update. It was the weekend after their last second Tuesday of the month in June. So they're doing this with the next update, second Tuesday of the month in July. So I think they're being as - they're reacting as quickly as they can, and this is something which is getting exploited a lot, so that's being closed down.

In other news, we'll remember last week we talked about this wacky report by the SSL researcher Ivan Ristic, whose company had recently been purchased by Qualys. He sent a couple tweets to me which came to my attention through Twitter. And then I thought, I wonder if he's posted anything in the feedback slot for GRC because I had no way to reach him through Twitter. Sure enough, he had posted a very polite note. He was a little bit tweaked by my characterization of him. And I think he actually said in his email to me, "No one likes to be called a moron."

So I was pleased with his note. And he showed me, in fact, I said, well, you know, it's a little hard to understand what it was you were trying to demonstrate. And I sent him a bunch of links to sort of confirming news reports, everyone picking up on this notion that only, what was it, 3.something percent of SSL certificates were valid, which was unfortunately the conclusion that people drew.

And it turns out that even Comodo, which is a very well-known SSL certificate authority, they put out a press release calling for Qualys and Ivan to please clarify what it was that he was saying because they felt it was grossly mischaracterizing the nature of SSL and what the industry was doing. He has done some blog posts since which attempt to explain his way out of this. The problem is, for example - I quoted from his blog post. At one point sort of in summary he says, "The reason we have so many domain names that do not have proper SSL certificates installed is that most of them are not intended to have them. And that's like...

Leo: Yeah.

Steve: Uh, wait a minute. It's like, what? No. That's not - he still doesn't want to say that...

Leo: He doesn't get it.

Steve: ...that he, that, well, that scanning domain names to see if they answer with SSL and then comparing the name to the certificate means nothing.

Leo: It's a broken system.

Steve: Well...

Leo: It's crazy. It's not...

Steve: There are, as I said when I talked about it, I said, you know, I've got SpinRite.com aimed at the same IP as www.grc.com. Because if someone puts in SpinRite.com, I redirect them to www.grc.com. I mean, it's just - it's simple to do. And that way I don't have to give it its own IP. No one needs to connect securely to SpinRite.com because I just bounce them over to GRC. So if you check the - if you look up the IP address for SpinRite.com, well, it's the same as GRC.com. But you can only have a single SSL certificate bound to a single IP, and that's GRC.com's certificate is valid and bound to the IP for GRC.

So what he's doing is wacky. And I think, I mean, he clearly knows what he's talking about. I spent some time at his site, looking through his stuff. The guy understands SSL. He just, I think, sort of went off maybe prematurely...

Leo: And now he's embarrassed.

Steve: Yeah. And then there isn't any way really to dig himself out of this because this...

Leo: Well, yes, there is. He should say, oh, never mind. I was measuring something inconsequential.

Steve: Yeah. In a way that doesn't affect or hurt the Internet, that doesn't reflect badly at all on SSL or security certificates. And what's funny is he talks about 22 million, like, certificates; but only three, I think it is, or less than that, have ever been issued in the history of man. So, like, all of his numbers are wrong. So it's like, well, okay, I mean...

Leo: A personal message, Ivan. You have two choices. You can say "I made a mistake," and we'll say, "Oh, he made a mistake, honest mistake, and he's admitted it." Or, if you continue to try to prevari- you know, kind of circumnavigate this, the assumptions going to be it was link bait; or he was in some way trying to get attention; or, you know - you have two bad choices. But one's a lot better, which is to say, oops, I made a mistake. The other choice is not good.

Steve: Well, and the fact is, he's not a moron.

Leo: No. Well...

Steve: He just, well, no, he's not. I mean, he really understands SSL. He's got that nailed. But I think he's trying to demonstrate something, or maybe someone said come up with something for the next security conference. I think he's speaking about this at the Black Hat conference.

Leo: Oh, okay.

Steve: And so he, like, thought, oh, I'll do that.

Leo: He made a mistake.

Steve: It's like, yeah, okay.

Leo: Big deal.

Steve: Speaking of making a mistake, it turns out - remember we talked recently about Firefox has been releasing versions pretty quickly? They went from 3.6.4; now they're at 3.6.6. Well, one of the things that they did with 3.6.4 was to add protection, and we talked about this a couple weeks ago, for frozen plug-ins. If Flash or Silverlight or - Flash, Silverlight. What's the third big one?

Leo: Flash, Silverlight...

Steve: Flash, Silverlight - there's another big one. And the third big one. If any of those three hang, well, it will no longer hang your browser. So what happens is...

Leo: Java?

Steve: Uh, I think it's...

Leo: No?

Steve: No.

Leo: You think it's a plug-in of some kind.

Steve: It's a plug-in. It's a big plug-in. Might be...

Leo: Shockwave? QuickTime.

Steve: QuickTime. It was QuickTime.

Leo: QuickTime.

Steve: Yes. Thank you, Leo.

Leo: Thank you, chatroom.

Steve: Flash, Silverlight, or QuickTime lockup. If they freeze, become unresponsive, then Firefox will see that and not have it hang the whole browser. You'll just have to reload that tab. Well...

Leo: Another Chrome feature, by the way.

Steve: Their - yes. Their timeout was 10 seconds. And it turns out that on older, slower machines, users playing one of Leo's favorite, I don't know if it's a game so much...

Leo: Plants vs. Zombies?

Steve: Farmville.

Leo: Farmville.

Steve: Farmville is hosted, apparently aggressively, in Flash. And so if you're playing Farmville on a browser, you're in Flash.

Leo: Right.

Steve: And it turns out that it's not uncommon for Farmville to make Flash unresponsive...

Leo: [Laughing]

Steve: ...for as many - for more than 10 seconds.

Leo: As anybody who's ever played Farmville knows.

Steve: So what happened was, when everyone upgraded to Firefox 3.6.4, their Farmville started crashing because Firefox was not patient enough.

Leo: Oh, yeah. Timeout, the length of the timeout is very critical.

Steve: Yes. Well...

Leo: How long you wait before you give up.

Steve: Which is annoying in itself. I mean, the fact that we have something so...

Leo: Nothing should hang that long.

Steve: ...random.

Leo: Yeah.

Steve: Well, true. Anyway, so the reason we've had this quick jump from 3.6.4 to 3.6.6 is that the Mozilla developers realized that a substantial portion of their user base - now, not as a percentage of all users, probably, but still enough that they were realizing, we're going to drive people away from Firefox to some other browser, that is, the Farmville addicts, unless we fix this quickly. So they quickly changed the timeout to - I know it's big now. I think maybe it's 45 seconds? They decided to go way high because they didn't

know if 15 would be enough, or 25, and they didn't want to keep marching it forward slowly. So they went to 45 seconds. I think that was the number.

Leo: Could Farmville, or a similar situation, could they send out a heartbeat of some kind just to say, hey, I'm thinking here?

Steve: Well...

Leo: Wouldn't that be the preferable thing to do than just be unresponsive for 10 seconds while the server is processing or whatever?

Steve: It's certainly the case that, in Windows, apps can become unresponsive if the thread which is servicing the user interface gets busy or hung somehow.

Leo: And no heartbeat would - that would block everything, including a heartbeat.

Steve: Right. And, you know, that often, you know, in less than well-engineered apps, let's put it that way. So, for example, in my DNS Benchmark I have a thread which runs the UI which is completely separate from the many other threads that are busy running around querying servers and doing other things. And so you can, like, stretch the window. You can be doing anything you want to, and everything keeps running. So it never is unresponsive. But any Windows users have run across the phenomenon of Windows itself putting up in the browser, I mean, putting up in the window title that this application is no longer responding, and you have the opportunity of closing it or waiting some more. So...

Leo: So I guess the question really is why the Zynga is hanging. If it's hanging because the Zynga database server is unresponsive, then Flash could say, well, I'm waiting for the database server, and Firefox wouldn't give up. But if it's Flash that's hanging, of course Firefox - there's no way Flash could say I'm waiting for something, it's just hanging up.

Steve: Well, and I'm not a Flash developer, so I don't know if they have the notion of threads and...

Leo: I bet you Flash is a single thread. I would be surprised if it's more than one thread.

Steve: That could be. And that could explain it. If it's waiting for the server - or it was related to slower machines. So it might just be doing a whole bunch of crunching to decide if it's time to plant more daisies or whatever the hell it is you do on the farm. So...

Leo: Yes. You hit it. You hit the nail on the head. You plant daisies. Push up daisies.

Steve: And in other Firefox-related news, I thought it was just interesting to note, and you probably saw this, Leo, that IBM has formally switched to Firefox.

Leo: Really. From...

Steve: Yes, the entire corporation.

Leo: ...Internet Explorer?

Steve: IBM has said, we're 100 percent moving to Firefox.

Leo: Wow. Well, welcome to 2008, IBM. You're gonna love it. It was a great year.

Steve: Yup. There was a little blurb about YouTube having some problems with cross-site scripting. But I think I saw you talking about it on one of the other podcasts.

Leo: Yeah, XSS, we were, yeah. In my very ham-handed way. I wish you'd been there.

Steve: Well, we've covered it in detail in one of our prior podcasts. The problem is, it is extremely difficult to allow the world to post its own text onto your web page.

Leo: It's hard to sanitize.

Steve: Yes. A cross-site scripting flaw is one where essentially, for example, in any kind of a blog where you accept comments, where users are able to submit text which the web server then posts onto the web page. What happens, of course, is that everybody who views that page is looking at the text that's been submitted. The problem is that hackers are never-ending in their creativity, figuring out ways to obscure the look, the actual content of what they're posting, in a way that, you know, I mean, whether it's scripting or using Unicode that expands to other characters, it's just amazing how sophisticated they've become in figuring out ways to post something which ends up being malicious or mischievous. In this case it was people getting a whole ton of - YouTube viewers getting a bunch of pop-ups, and some redirections to adult websites. So in this case not anything really nefarious. But...

Leo: Well, and they fixed it. Google fixed it.

Steve: Yes, yes. They immediately took the comments down and then looked at how it was being done, added some filters for it, and then put them back up. Which is really all you can do.

Leo: I went to a great presentation by Dan Kaminsky - and by the way, I said hi, and we love you, and thank you for finding that DNS flaw - on exactly this, on what the best ways are to sanitize inputs. Because this is really one of the biggest security vulnerabilities is sanitizing user-entered data, whether it's a URL or comments...

Steve: In Web 2.0 that's the problem.

Leo: It's the problem. And he came up with a very interesting solution. And I didn't fully understand, but it was a great session. He base-64s everything. And I'll see if I can find it. He'll probably be giving it at Black Hat or somewhere, and I'll see if I can find it. But it was a very intriguing solution. And he gave a lot of different scenarios and so forth. And but the point not being let me suggest a solution because I don't understand it well enough to actually suggest it. But the point being that even the top security people, this is what they're working on right now is how do we have a - let's create a library that we can safely sanitize user input. This is a big issue.

Steve: Yeah, yeah. A little blurb from India. The Indian government, stating their need to monitor for terrorism, has demanded from Skype and RIM - the people who do Blackberry - and Google the ability to read, essentially the ability to decrypt their content. So the India government says, as an antiterrorism measure, all the text in Skype and Blackberry and Gmail need to be decryptable by the government.

Now, this actually came up in 2008 with RIM and the Blackberry. And it was interesting because I learned something I didn't know before, and that was that RIM's architecture specifically prevents this. Only the end users have the keys necessary for communicating. So it was designed in a - my favorite acronym - TNO, Trust No One mode so that they said, look, we're not sure we'd want to comply, but we can't. Our system is so secure that we don't have the keys to decrypt our customers' communication. The architecture doesn't allow it. And the issue just sort of, in 2008, when this was brought up between India and RIM, it just sort of died off, and no one's really sure what happened. But RIM's architecture, as far as we know, hasn't changed and is still super secure. So this just sort of represents a worrisome blip on the radar that, I mean, you can understand governments feeling the way they do. But people also want to have the right to communicate privately.

Google is expanding its suspicious log-on location technology. We talked about that many months ago, where Google will notice if the location you log on from appears to be geographically very different in a very short period of time. So, for example, you log in in the USA; and then a short time later, like half an hour later, you log in in the UK. And they say, uh, wait a minute, how did he get to the UK that quickly?

Leo: I've had that happen. And it's a little disconcerting, but I'm glad they do it.

Steve: Yes. It can misfire because geolocation by IP is not a perfect science. And they don't do anything except notify you. That is, they're not locking you out of causing a big problem. They had only been doing it for Gmail. And the cool thing is they're now expanding that, sort of using Gmail as an initial test bed and deciding that this ended up being a good thing. They're now expanding it to work across all of their web-based services. So that's very cool.

Leo: I think they deserve huge applause for what they've done with Gmail. And I hope that they extend this to web services. Maybe this is what you're talking about, but if you go to the bottom of Gmail, you can see all the IP addresses that have recently been used to access your account.

Steve: Yes. And that is now going to be universal across all Google services.

Leo: Oh. That's such a great idea because, if you have any suspicion, and I have from time to time, that somebody has got my password or whatever, you can immediately verify that. I just wish they went back farther in time. It only goes, I think, 24 or 48 hours. It is so incredibly useful to see that and know exactly who's been accessing your account.

Steve: Yeah. It totally makes sense. Now, there were in the news in the last couple of days some stories that were a little overheated, saying that Skype's encryption had been cracked, people worrying about that. There was, essentially, some very good reverse engineering done. And I wanted to put everyone's mind at rest that Skype's encryption has not been cracked. So a guy who people believe is using a pseudonym of Sean O'Neil essentially posted that they had reverse-engineered Skype's encryption.

Now, Skype's overall cryptography architecture is really good. I've looked at it in the past just because I was curious, and I've got it on my own notes here to do a whole episode on the security architecture of Skype, much as we're going to talk about the security architecture of LastPass in this episode. We have developed over the last five years, four weeks short of five years, all of the bits and pieces we need to, where there we're talking about specific technologies. Fitting these together into systems is one of the things I want to spend some time on moving forward, and looking at how Skype works in terms of how do they use these things in order to create an architecture is something I want to do.

Well, what this Sean O'Neil, whatever his real name is, going by the name Sean O'Neil, what he, or they, have done is essentially they reverse-engineered some pieces of what Skype has done which Skype has kept proprietary. Now, essentially, we've talked about security by obscurity. That's a favorite term people like to use sort of in an absolute sense of, well, you know, if you rely on security by obscurity, then you have no security. It's certainly the case that, if you depend upon obscurity, you can't ever count on it enduring.

Well, Skype is not depending upon obscurity except to prevent competitive creation of clients. That is, Skype-compatible clients. So back in the early days of this podcast we talked about the RC4 cipher. RC4 is the cipher which Skype uses because it is extremely lightweight, extremely efficient, and, if you use it right, extremely secure. RC4, of course, popped up onto the world's radar because that was the stream cipher chosen for all of those reasons by WEP, the insecure WiFi encryption which was originally developed. The problem was not that it was using RC4, but it was misusing RC4. It was not using it in a secure fashion. And it was exactly that that caused a lot of the Achilles heels of the implementation of early crypto on WiFi.

Well, the Skype guys know their crypto. And they're using RC4 the way it should be used. But they haven't published the details of the keys and the initialization vectors which are part of what you have to have in order to create a Skype client. They've not

wanted people creating knockoff Skype clients. If someone did, then that clone of the Skype client would still have to obey all of the security architecture, which would mean it would have to be as secure as Skype's own client. But it was the fact that there was a lot of this that was unknown, like the real deep inner plumbing, the Skype folks had never published. And they probably - they'd apparently wanted to keep it to themselves so that people could not create Skype clones.

Well, now this is out. These guys reverse engineered it. And one of the other major principles of this podcast that we've talked about from time to time is of course it can be reverse engineered. Anything can be. It's why it took no time at all for Blu-ray DVD and HD DVD, those ultra state-of-the-art cryptography technologies, to get reverse engineered, because they were going to be - if it's going to live in someone's living room, and the device that's there can decrypt it, and it has to in order to put it up on the screen, then engineers are going to figure out how it's working.

Similarly, nothing that the Skype folks can do by definition can prevent someone from reverse engineering the guts of that client. You and I are using Skype right now. Our clients at each end are encrypting and decrypting our conversation on the fly. So the fact that I've got something running in a computer right here next to me which is doing that means someone can look inside while it's doing it and figure out how. There just isn't any way to prevent that.

So this is news from the standpoint of, well, yes, something that the Skype folks had so far managed to keep to themselves, to keep proprietary, but which their security architecture does not depend upon at all, that's now been figured out. It's been reverse engineered. So, okay. That doesn't in any way weaken Skype security. It means that you could, if someone wanted to, create...

Leo: Maybe.

Steve: ...knockoff Skype clients.

Leo: It's easy enough for them to change their secret sauce, you know, and make it secret again.

Steve: Right. Right. Oh, that's a very good point. They will - well, yeah.

Leo: They control the client and they, I mean, there's...

Steve: Yup.

Leo: Update it.

Steve: Yup. They could add some ciphers. And if it's supported at each end, then the new cipher would take hold, and over time Skypes would get updated, and they would all support the new one, and then it would be back to reverse engineering, so.

Leo: I am still, as good as Skype is, remember you had a great project that you kind of thought about for a kind of better way to do it, for podcasters only, with a ring buffer and a - I want to find an open source programmer who knows telephony to write something that's just for the kind of stuff we do. There's a lot of features in Skype we don't need. And there's features that we need that Skype doesn't do.

Steve: Yeah. Um, uh [laughing].

Leo: I don't want to assign this to you. I know you've got other things to do. In fact, CryptoLink is so much more important, I wouldn't even dream of saying anything. In fact, forget I mentioned this.

Steve: Well, yeah. The idea, just in case our listeners are curious, was that there's no reason that you couldn't have absolutely perfect communication by having another channel running in the background which makes up for lost packets.

Leo: Right. We need real-time signaling for conversational reasons.

Steve: Yes.

Leo: But we would like...

Steve: A perfect recording...

Leo: ...a perfect recording.

Steve: ...result.

Leo: So if any packets are paused or delayed, we would like to preserve them and use them. Either you record it locally - actually, ideally, you record a copy locally, we get a second channel with a perfect copy, and then we have the channel that's on the live stream that we use for our conversation that's real-time.

Steve: Yeah.

Leo: But imperfect. I think I'm going to - this would be a great project for TWiT to fund. I would make it open source because I think other podcasts could really use this. Skype has been great. But Skype is designed for something completely different, and we're really not using the tool for the purpose we need. And I can think of a few other things like multichannel audio that would really be nice.

Steve: Yeah.

Leo: So I don't know. Maybe that's a crazy idea. But I'm going to talk to some people.

Steve: Complex, though, when you add video.

Leo: I know. It's always a problem, isn't it.

Steve: If it were audio only, it's like, eh, it's not a hard problem. I mean, I could give somebody all the work I did. Because I did, you know, develop that codec that you listened to up in Toronto once that was actually doing it. And in fact I used it to remember exactly how to do that Star Trek phrase backwards. The way I wrote it, you could reverse the buffer, and it would say whatever you said backwards. So that was fun.

Leo: It's fun. Being a programmer - to follow up on the dog killer, portable dog killer, I know we were talking about physicality, and it's fun to build things, and we're going to go to Maker Faire at the end of the month and see people who are doing amazing things with their hands. But I think programming is also an incredibly satisfying hobby. And when your little things like this - to be able to write your own software is so great and so satisfying. And you know what, in this world, potentially very lucrative. If you're a young person thinking about a career, learn to program. If you've got aptitude for it, the world needs you. I need you.

Steve: Well, and I don't know what's going on, but there seems to be some sort of acquisition fever happening lately. I mean, everybody's buying everybody.

Leo: Yeah.

Steve: And one of the things that I'm seeing is that, you know, if you develop something which is good, big companies will come along and just buy it because it's easier for them to do that than to develop it themselves.

Leo: Yeah.

Steve: So you mentioned, it was on, boy, I think it was one of your - I think it was on The Tech Guy last weekend, in the context of Starbucks WiFi now going completely free.

Leo: I meant to have you on, and we've got to still do this on the radio show, get you to call in. How do you secure yourself now that millions of people are going to be using Starbucks' free WiFi?

Steve: It's funny, too, because I noted when I went back for my refill yesterday there

was clearly more, I mean, I'm very much in tune to the pulse of my local Starbucks. There was clearly more businessman/executive sort of looking dudes with their laptops...

Leo: Yeah. Can you imagine.

Steve: ...than we normally see.

Leo: What a fertile ground for a hacker.

Steve: Well, it is. And I thought you did a really good job, actually, by following, I think it was Lifehacker had a nice story about the things you needed to do.

Leo: Yes, I used their outline, yeah.

Steve: The one thing that I have said before, that occurred to me as I was listening to that list, that was sort of missed is that one of the things that Windows has done that I think is very nice and clever, not original with them, but still a good idea, was the Windows Firewall, it allows local filesharing on the LAN, that is, it looks to see whether the source and destination of filesharing traffic is in the local subnet, or whether it's an IP bound for outside the local subnet, which it blocks. Which is really a nice solution because it automatically means that your own residential or small office filesharing, which is so convenient to map drives and just - or map folders and just drop files on them, and they shoot across the LAN, and they pop up on the other machine. That's really handy. And then you're automatically protected from that not going global because it looks to see if it's on the LAN.

Well, the one big problem with that is that, when you're on a WiFi network, that network is your LAN. Which means that you're in, suddenly, you go to Starbucks, and you're taking a machine, a laptop, where in your home network you've got things, files and drives file-shared, made available. Well, when you go to Starbucks, what the firewall now sees is all the people at Starbucks, whether they're good people or not, are on the same LAN. So that technique, which is very useful when your LAN is friendly, works against you, or certainly not for you, when your LAN is unfriendly. And open WiFi is by definition an unfriendly LAN because you're not encrypting your traffic. And suddenly, I mean, without intending to, you're sharing any of those resources on your LAN, which is now the WiFi network. And that's something I've never heard anyone mention before, so I just wanted to - I have said it on this podcast before in that context also because I always remember, it's like, oh, you know, whoa, be careful about that.

Leo: Good point.

Steve: Because that's potentially a biggie.

Leo: But you should absolutely turn on the Windows Firewall.

Steve: Oh, yeah, it ought to be on all the time. It's not in your way normally, so...

Leo: Exactly, exactly.

Steve: So that's a good thing.

Leo: But it's just not enough.

Steve: Right. And I've got a great little SpinRite story to relate. And then we'll get into our content.

Leo: Good.

Steve: This is from Paul Bye, and he emailed it to our tech support address, so I don't know where - oh, there, it's Rochester, Minnesota. He said, "Dear Steve and GRC staff, I'm sure you get so many of these you can't keep up." But we love trying to keep up. "But I wanted to add my praise and thanks to you for SpinRite. I've been a faithful listener to the Security Now! podcast and was just itching for a reason to buy SpinRite. I finally had the chance when my mom's hard drive crashed last year." When did he send this to us? Because, oh, 2nd of February. I'm digging down a little bit. When it crashed last year.

"The power in her house has always been suspect, and it finally did her hard drive in. While the drive was really beyond repair for future use, I was able to run SpinRite and recover all her data, including her precious pictures of her granddaughters. She's now set up and knows how to run regular backups.

"The other day, one of my heavily modified, dual-drive TiVo systems started making funny noises. I immediately unplugged the machine, knowing all I needed to do was run SpinRite, and it would probably be okay. While not containing any crucial data, I had many hours of recordings and many, many hours of modifications I had made" - that sounds like my TiVos, too - "I had made, and really would not have wanted to lose them." Amen.

"Three hours later, 1.5 hours per drive, it was up and running, no funny noises, in fact, quieter than it had been before. And maybe even a bit faster. Besides not having to redo all the work I did on it, the savings of buying a new drive alone paid for SpinRite. As a programmer, I certainly appreciate how few pieces of software there are that live up to their billing. SpinRite is definitely one that does. I love the podcast. I think it's the best on the Internet. And I thank you for all the work you do and share with listeners. Thanks. Sincerely, Paul Bye, Rochester, Minnesota." And thank you, Paul.

Leo: Thank you. We love people who buy SpinRite. GRC.com. Let's get to the meat. I'm very excited about LastPass. I think I found out about LastPass from this show. I think somebody wrote a note. I've used RoboForm, which was Windows only. I think they're doing a Mac client. And I've also on the Mac used 1Password, which was Windows - rather, Mac only. But then when I found LastPass, first of all, it's very

affordable. There's a good free version, and a buck a month you get some additional features. A buck a month. And it works on everything that I use, including all of my portable devices, the iPad, the iPhone, the Android systems, Blackberries. So I fell in love with this. But I'm not you, Steve. And I just trusted. I had to say, well, I trust that they're doing it right. It looks like they're doing it right. But I'm very glad that we're going to find out. The Steve Gibson check-out.

Steve: Okay. So one thing I want to do in this podcast is I'm going to go against something that we've done in the past, which is to rely heavily upon previous podcasts.

Leo: Okay.

Steve: For explaining the crypto stuff I'm going to - I'm not going to assume any prior knowledge. We're not going to go into the depth that we have before. So people who want to follow up, who, like, say, well, I want to actually understand how some of these things that Steve talked about actually work, we've covered all of this in the past. So that's all there. But I don't want to assume that someone listening to this remembers all of that. So on the crypto stuff there'll be a little bit of redundancy that way, but not a painful amount of it. And the reason I say this is understanding the architecture that these guys developed is the key to understanding why it's safe to trust them; why I trust them; and why I've completely switched my entire solution for managing passwords, after spending days researching it and testing it and playing with it, over to LastPass, which I have.

So let's step back a little bit and understand what the problem is we're trying to solve. The very early episodes of this podcast, nearly five years ago, we spent several episodes talking about passwords, personal password policies and the whole issue. And the password is sort of the original security technology. I mean, back dating from the early days of UNIX machines, which were the first machines on networks, it became important, became crucial, necessary, to identify users. And so the idea was you'd have a username, and you'd have a password, the idea being that the username was something everyone knew, it was public, but the password was something only you knew.

And we've talked what it takes, the whole problem of managing passwords. The problem is that all other things being secure, that is, assuming that a system of some sort, whatever it is, doesn't have any other security problems, if it's password-based, then the one vulnerability is guessing the password. That is, if you know someone's username you can think, okay - I mean, and we've seen this in movies and things. It's like, okay, let's see, what might their password be? Well, I know the name of their kid, so let's try that. I know the name of the dog and their parents and...

Leo: The dog is what happened to Paris Hilton. Everybody knew her dog's name.

Steve: Right. Not a good password to use...

Leo: No.

Steve: ...for that reason. So the problem is that the vulnerability is guessing the password. In fact, remember that just last week we talked about the FBI and the Brazilian government both failing, after years of trying, to crack the TrueCrypt encryption of someone whose drives they had acquired who was a suspected money launderer. They'd spent years using a dictionary attack where they had a dictionary of words that they just kept trying. Well, that was brute-force password guessing. This person whose drives these were was smart enough to use a password not in a dictionary. Had it been a simple dictionary-based word, his security would have been cracked that way. So the idea is you want a password that isn't going to be guessable, that isn't in the dictionary.

Well, the other thing you need is something - so we'll call it gibberish. It's 32X5707 or something, just gibberish. Now the problem is it needs to be long because the next attack on a gibberish password is to try every possible combination of gibberish. You start with "A" - and don't use "A". That's a bad password, by the way. That's the first one you - that's the other first one you try.

Leo: Or "Z" because sometimes they try the backwards way.

Steve: Well, and even if you did "Z," but they started at "A," that's only going to take them 26 tries.

Leo: That's a good point. So anything "A" through "Z," bad.

Steve: Bad, yeah. And then you go AA, AB, AC, AD and so on. And so now, if it's a two-character password, and assuming it was all, for example, lowercase, well, then it's going to be 26 squared. And we can still try that many in a short time. So the point is that it is possible in many scenarios to try every possible password.

Now, every possible password gets to be many pretty quickly. But what that means is the longer your password is, the stronger it is because every character you add - say that we have the characters lowercase "a" through "z." That gives us 26 possible characters in the so-called alphabet. Then say we had then uppercase "A" through "Z." And these passwords are case sensitive, meaning that it matters whether it's an uppercase "A" or lowercase "a." So now - so before we had 26 lowercase. Now we add 26 uppercase. So we're to 52. Now, if we add the digits 0 through 9, we're at 62. And say that we just add two more special characters, plus and minus, that is, we allow a plus symbol or a minus symbol, well, that gets it to 64.

Well, 64 is a special number because that's 2^6. Which is to say it's the same as six bits of password strength. So using just the alphabet and the digits, plus a couple more characters, we get six bits of strength per character, which is to say 64 possible combinations for a single character. For two characters, that's 64x64 because we have all of the possible characters with 64, and then 64 of those first characters for all of the second characters. And, if we had three characters, it's 64 times more. And the fourth character is 64 times more. Well, if you start multiplying 64s, this gets to be a very big number very quickly. So the point is that computers are fast. And who knows whether the FBI actually started trying gibberish. Maybe they did. But if it was sufficiently long gibberish, you just can't try them all.

Leo: Well, how many years did they try it?

Steve: Well, they tried for many years. I think it was two years.

Leo: Two years, if they have fast systems, is billions and billions of attempts.

Steve: Attempts, yes. Although systems are also, and I think TrueCrypt being a well-designed system, the other thing that systems will do, and I know for example that WPA, the good encryption for WiFi, does this, is that the actual algorithm for turning a password into a key is itself complicated. So I don't remember, I think it's 4,096 iterations of some cryptographic functions that WPA goes through. So a single attempt takes a while, but not long. It's short enough that we don't notice it. But what it means is that many, many, many, many attempts is scaled up by a deliberate increase in the complexity of the algorithm that gets the key from the password. And I would be surprised if TrueCrypt hadn't done the same thing. So that meant it was infeasible, that is to say, yes, you can try lots of passwords, but each one is expensive in computational...

Leo: It's going to take you a while, yeah, yeah.

Steve: Exactly, in computational time. And that's a deliberate overhead added by good security people who want to prevent this kind of brute force, just try every possible combination of gibberish. So what we've done is here in the last 10 minutes we've walked through the problem. The problem is we want a really long gibberish password if we don't want - in order for it to be secure.

Now, the next problem is, you know, you could imagine, okay, I'm going to come up with a really long gibberish password. That's going to be my password. But you don't want to use the same one all the time because the other problem is that, say you log onto your bank with your email address and this monster amazing password. It's, like, great. That's safe. But if you also log onto StoriesMyDogToldMe.com or something, with...

Leo: That's the problem.

Steve: ...your email address and that same monster amazing password, now the problem is, yes, that's really strong, except that we're not too sure about the security of the website StoriesMyDogToldMe.com, nor about the employees who work there. Because remember, they got your email address and this amazing monster incredible password of yours. So, and the same might be the case with malware in your computer. We know that people are getting themselves infected. If malware saw you log onto StoriesMyDogToldMe.com with your email address and this monster incredible password, there's nothing to prevent it from rummaging around in your computer and noticing that you're a BofA customer and thinking, hmm, I wonder if this person uses the same monster incredible password for all of the different sites they visit. And so you can imagine, if you were someone who had an amazing password, but you only used one, then think of the liability. Every one, all the sites you log into, whether it's Facebook or Twitter or TweetMe or StoriesMyDogToldMe and your banks and everything, they've got

your email address and your password. Which means it's really not yours anymore.

Leo: No.

Steve: I mean, everybody has it. And if there's ever a chance, I mean, they could just guess, you know, ah, I wonder if this person uses BofA? I wonder if they use Chase? I wonder if they use - they could just put in your email address and this password, and wow, it's going to work. If they guess even, like, where you might go, like what groups you're members of. So not only do you need a really long, incredibly gibberish-y password, but you need a bunch of them.

So now we've got a big problem because how do you handle that? How do you manage that? How do you - now you literally somehow have to write them all down or record them all. And this is a big problem. If it was sufficient to have one really monster incredible password, that's like, okay. You could potentially memorize that. We've talked about fun ways to do that, like think of the lyrics of a favorite song and choose the first character of each word in the lyrics in order to help you remember it. But then, you know, maybe salt it with a few digits in between, or if the lyrics had the word "four," use the number "4," that kind of stuff. So there are fun ways to help you with that.

But the problem is, that's not what you want to do because we already established that it's really not safe to use the same, or even a small set, of really good passwords among a huge number of sites because you can - because of the problems of inter-site or cross-site usage. So the way to be secure is to have long, gibberish-y passwords, and a separate one for every place you log in. That way no one can ever try to log you in, to log in as you, to impersonate you for whatever nefarious purpose. If they know your password for this site, doesn't help them at all on some other site. Which is really what you want.

The problem is managing that, which is what various password management tools do. LastPass does that. The idea is that LastPass has plug-ins for, that is, additional functionality that they add to all of the popular browsers. They've got browser plug-ins from everything from IE v6 on up, Firefox v2 on all platforms that Firefox runs on, Google's Chrome from v4 on, on all platforms where Chrome runs, Safari from v3 on OS X and from v5 on the PC, and even Opera, which is plug-in hostile.

Opera doesn't have a plug-in architecture. So they use something called "bookmarklets," which we'll talk about a little bit because that's one of the solutions, for example, over on a browser like the iPad, where it also doesn't allow plug-ins. They actually have a clever solution. They have their own tabbed browser which is, I mean, LastPass has now an iPad tabbed browser which includes the LastPass functionality as a way to get it.

Because here's one of the reasons I like LastPass so much is they've just completely covered the landscape. For mobile devices they've got iPhone, iPod Touch, Apple's iOS. They have the tabbed browser for the iPad. They've got their plug-in for Android and for the RIM Blackberry, for Windows Mobile, for Symbian and for Palm's webOS most recently. So it's everywhere. And that's really what you need because essentially what we're going to do, what the security-conscious LastPass user will do, is go through and probably improve your passwords. Probably there's been some laziness along the way. There's been the reuse of passwords that you like or that you've memorized, just because some new site you're visiting for the first time says what password do you want to use? And so you go, ahhh, I think I'm going to use the one that I like.

Well, okay. That's a danger, as we have established. So you really need to come up with and to even fix retroactively passwords which you're using which are not safe. The problem is, how do you remember them? And that's what LastPass solves. However, if it wasn't available on anything you might possibly want to log in on, now you've got another problem because you've got these long, gibberish-y passwords that you can't possibly memorize, which is part of why they're so good. But unless they're available on any platform you would be using, you've got a problem. So they've got the bases covered, I mean, absolutely and completely.

Leo: Yeah. I mean, I use everything there could be used under the sun, and I haven't found anything that it doesn't work with.

Steve: No. It's always there.

Leo: Even the iPad.

Steve: Yes, even the iPad. What these bookmarklets are, a bookmarklet is a bit of JavaScript which is like a URL, that is, it sort of runs like script on a page. And that allows them to sort of shoehorn themselves into literally any browser. So if you didn't have any plug-in, or for example you were using somebody else's browser that didn't have a plug-in, you could still use these bookmarklets in order to get access to your own personal library of passwords. So that's what LastPass creates is your own personal library of passwords.

What LastPass users have a level of reasonable discomfort with, and I did when I was first installing this and setting things up, LastPass has also a form fill-in capability. And it was suggesting, why don't you give me your credit card numbers? It's like, uh, what? And it even has a secure vault where you can put just your own notes which you want to have available anywhere, that is, on any of these platforms, containing anything whatsoever. The question is, how is this safe? How is it that I am not giving the LastPass people, who I want to trust, but do we trust everyone who works there? Do we trust everyone who has ever worked there in the past, who will ever work there in the future? Do we trust that, like, that somebody won't break into their servers in the middle of the night and have this huge massive win of getting all of the usernames and passwords for everyone who is using LastPass?

So the way this works is, the reason I'm using it, is I now understand how it works and why it's absolutely trustable, is that very much like Jungle Disk, which we've talked about in the past, all the encryption is done locally. That is, at no point does LastPass receive anything other than what looks like a block of pseudorandom noise. We've talked about how, when you take so-called plaintext, the normal readable, human readable, your username as an email address and your actual password, and you encrypt it with a good cipher, it turns it into, under the influence of a key, which is the key to the whole process, under the influence of the key, it turns it into noise, absolute pseudorandom bits that mean nothing.

So that's what the LastPass system gets and saves. It is absolutely no use to anyone because they never get the key. And they've gone to great lengths to arrange never to get the key. When you log into their system, you do so with your username, which is your email address, and your password. That's put together, it's concatenated into one long string. They sanitize the username a little bit. They lowercase it, and they remove

the so-called white space, you know, spaces and things. That just makes it a little more robust. The password they don't change at all. So that remains case-sensitive, and special characters and things can be in there. They leave that alone.

But, for example, email addresses are not case sensitive. You can change the case in an email address. And so since they're using their email address, people's email addresses as their password, users might not be careful about the case in their email addresses, so they make that case-insensitive. They always lowercase the email address ASCII characters, the alphabetic characters. So they put all this together into one blob. Then they do something called a "hash." They use SHA-256, which is a - SHA stands for Secure Hashing Algorithm. The listeners that have been listening to the podcast for years know what that means.

For people new to this, a hash is what's called a one-way function. You can take any amount of text or anything, binary data, anything, any amount of data, and run it through this process called "hashing," which always results in a fixed-size thing, sort of a fixed-size token. And what's unique about this is it is "computationally infeasible," is the technical jargon that cryptographers use, to go the other direction. That is, it's very easy to put stuff into this - think of it like sort of as a meat grinder. But it's impossible to ungrind the meat. It's been ground up. It's been completely - it's been turned into this 256-bit result such that anything you change in the input changes everything about the bits in the output. Yet anybody, no matter how much they want to, no matter how much they look at it, they can't go the other direction.

So the idea is that when you log in, when you give your system your LastPass username and password, the first thing it does is it runs it through this SHA - it lowercases the email address, removes the white space, adds the password, and then it does this hash to it, turning it into a 256-bit blob which tells the blob holder nothing about your username and password. It's just like it's been digested into this thing. In fact, hashes are called "digests," also, for that reason.

What that is, is that is your cryptographic key. That's the key which your system will use, both to encrypt your data which is being shared with LastPass Corporate, and also to decrypt it when LastPass Corporate sends this back to you. They're holding the encrypted results of your own personal database, just because that's what they do. That's the service they provide, essentially, that and creating all these amazing plug-ins for everything anyone's ever heard of. So but what they're holding, they have no ability to decrypt. They never get the key. That never leaves your system.

Now, they do need to know that it's you. That is, they need to know that it is you who are logging in. And so there needs to be an authentication process, so you identify yourself to them. But we don't want them to get the key. So what they do is, they take that key, the cryptographic key, and they add your password to it, that is, they concatenate your password to your cryptographic key, and they hash that. So they do another one-way function on your crypto key with your password, which they don't know because they never get it. But they get another blob.

So this second blob, this second output from the hash, that's your unique ID. That is, the only way to get that is if you take your username and password, hash it, then add the password to that and hash it again. So it absolutely depends upon both of those pieces of information. So then your username and that goes to LastPass to identify you. And because that contains your password twice hashed into it, nobody who doesn't have your password, even if they have your email address, is able to produce that blob. So you have to have your email address and your password run through this hash twice to get that blob.

But notice that your cryptographic key, which is sort of the first byproduct of that because that's the output from the first hash, that goes into the second hash but is lost in the hashing process, thanks to it being mixed with your password. So the LastPass people never get your crypto key. They get a different unique token that identifies you to them so that you're able to log on securely to their facility. And these guys are so paranoid that they don't even save that on their servers. They don't even save that special logon blob, the output from that second hashing process.

Instead they, at the time you create your account, they come up with, they use a random number generator at their headquarters to create a unique 256-bit token which they save with your account. And whenever you're logging in, they take this 256 blob you're sending them that's the result of these two hashing processes. They add that to this unique 256k random number, and they hash that. And that's what they compare to what's stored with your account. Which is to say they never store that logon token. They store the result of hashing that logon token with a unique 256-bit value that they created for you. So they dynamically see if it's the same, but they never save your logon token. They just - they don't want it. They don't need it. So they're able to perform a dynamic check whenever you need to authenticate, but they don't keep it statically.

So, I mean, this thing is secure every way you can imagine. And it's simple. The reason it appeals to me is that there's no hocus-pocus, there's no mumbo-jumbo, I mean, I can explain it to you and understand it, which means I believe it. Because there's no, oh, then a miracle happens, and just trust us. That's not necessary. The result of this 256-bit hash where they take your username and password and hash that to get the key for the encryption, that is used with the industrial-strength, maximum-strength, AES 256-bit cipher that we've talked about, which takes 128-bit blocks at a time and turns it into 128 bits of gibberish under the influence of the key.

So the whole concept here is that we establish a database of domains that we're logging into, and usernames and passwords for those domains. And this is our personal database. And the beauty of this, and I've been playing with this now for about a week, is that, for example, I did change a couple passwords because I'd been a little lazy, too. And I thought, okay, now's the time. So I changed those passwords here at home on my system in Firefox, and changed them in the website. And LastPass watched me change them. I said, okay, remember this. And LastPass remembered it.

And then the next morning on my iPad I wanted to log into the site. Well, I didn't write it down. I mean, you can't write these things down. Well, you could, but it would be a pain. Using my iPad, and I don't remember if I was using the bookmarklet for the iPad which is easy to create, and I have, or LastPass's own iPad tabbed browser which they have available. But whichever, I opened the site, went to the logon page, LastPass saw - oh, it was the tabbed browser because it was an automatic process. The bookmarklet, you invoke it to fill in the form. It won't do it for you automatically. When you're using any of these plug-ins which are so widely available on virtually any browser that allows a plug-in, they've done that. And this is all cross-platform - Windows, Mac, and Linux - all of this stuff.

So it automatically saw that I was at the logon page, populated the form, and hit login button for me. So the whole process was automatic. I mean, frankly, I've been spoiled now in the last week because this thing works so well. And my point was that, because this exists in the so-called "cloud," in the Internet that we're all connected to, the change that I made in the logon credentials for that site, whatever it was, I don't remember now, it was stored by LastPass. The plug-in resynchronized itself with LastPass Corporate, and they're on several continents and several different data centers. They back up themselves locally, and then they back up using Amazon's S3 service nightly so that

that's all being kept safe. And we'll talk about what happens if they go away in a second. And then the next morning on a machine I had not used, on a platform I had not used, I was able to log in seamlessly using these new credentials because it was synchronized through the Internet. I mean, it's absolutely perfect.

Now, now we've established this fantastic database, different passwords for everything. But we're dependent upon it. We can't function without it because we're no longer using something simple that we've memorized, or we're no longer using something complex, like our one master galactic password that we're using everywhere, because we know that's not safe. But now we've become utterly dependent upon LastPass. I mean, it holds the login jewels to our entire online existence. So is that safe? I mean, can we depend upon it? Well, we don't have to. They have covered that base, too. They have a standalone executable, a standalone gizmo. I'm trying to think of what it is they call it. Not Sesame, that's their one-time login deal. I've got it written down here somewhere. Maybe, oh, LastPass Pocket.

Leo: That's right.

Steve: It's called LastPass Pocket, available for Windows, Mac, and Linux.

Leo: I've never used it, but that's another feature I think is really important.

Steve: And I have used it. What this thing does is it is a standalone personal database decrypter. So you can, using their web interface, or using any of these plug-ins, you can say, "I want to dump my database." You can, if you want to be risky, you can dump it using one of these plug-ins as a CSV, a comma-separated value, plaintext file. Meaning that what you will see is all of this data, the domain, the username, and the password that they're holding for you, in cleartext, in plain ASCII. I say that it's dangerous because that would be like a feast for any malware that happened to be on your machine. I mean, there in pure readable simple text file is all the way to access all the things you ever access. What's cooler is, I mean, but you can do it if you want to, and then copy it to a CD or do whatever you do with things that are really vulnerable.

What's cooler is you can dump it in its native form, in that encrypted blob that only you know how to decrypt. And you can do it anytime you want. So I would say, for safety, every few days or actually anytime you, like, make major changes to this LastPass database. It's as simple as clicking on the little LastPass button on your browser and say "dump." "Export" is the word they use. You export this entire blob of encrypted data to your drive.

Now what's cool about it is that the LastPass Pocket app, it doesn't install, no setup, it's just a simple "it just runs" executable, which is the kind of EXEs I love, or executables that I love. You are able to give it your username and your one LastPass - your LastPass password, which is the way this whole system functions. And then it's able to decrypt your backed-up blob and allow you to view it, to see it, to browse it.

So in addition to everything else, this is trivial for you to carry with you. You can stick it on a thumb drive or on a CD or leave it lying around. It's just a blob of gibberish. It is absolutely invulnerable to attack. It's the same content that they're storing for you. It's no longer synchronized in the cloud, of course, because it's a now a standalone file. But what it does is it gives you the security of not needing to depend upon dynamic

connectivity to the Internet. Now, the plug-ins do store a local copy of this so that, if you were offline, the plug-ins in browsers have their own most recent copy. So you're still able to access them locally and look at the data.

Now, the next thing we need to look at is the vulnerability to somebody impersonating you logging into LastPass. Because notice what we've done now is we've created valuable content. It's now safe with the LastPass folks. It's now backup-able. We're able to clone it to create our own local copy. And we've got viewers that run on all platforms so we know that we'll be able, if the worst happens, to manually look up, gee, what is my password for Amazon.com?

And I should mention also that there is a secure password generator as part of LastPass, as part of the plug-ins. You can customize it a little bit. You can tell it how long you want your passwords to be. You can say I want it to be - I want uppercase A through Z, lowercase A through Z, the digits. I want to allow or not allow special characters. I want to require a certain number of digits to be in every one of these, which gives you maybe a little more security. And also, of course, you're able to specify the length, anywhere from three to a hundred characters. Well, don't use a hundred characters. First of all, many sites won't accept a hundred-character password. I would say, and I thought about this for a while, 10.

Leo: Yeah, I was going to ask you, this is a great question.

Steve: Yeah, I would say...

Leo: Ten, ten is enough. If it's random. It's upper, lower, punctuation and numbers.

Steve: Yes. If you were to use, well, even without special characters. I'm a little hesitant about special characters because some sites will, like, barf a little bit if you use funky random circumflexes and tildes and vertical bars and things. So let's return again to upper and lower case, which gives us 52; the digits, which gives us 62. Well, 62 is not quite 64. It's obviously two short of 64. Well, that happens to be, don't ask me why I know this, 5.94 binary bits of equivalent strength. Not quite six bits, 5.94.

Leo: It's one of those random things you just had in your mind, isn't it.

Steve: Eh, something that you cryptographers know. So if we take 5.94 - I've got my calculator in front of me - we multiply that by 10. Oh, that was easy. Why did I multiply? Why did I get my calculator? That's 59.4 - oh, I know why I got it, it's for the next operation. 59.4 bits, equivalent bits of binary strength. And so if we raise 2^59.4 - I didn't do it. Two to the power...

Leo: Do you have a calculator in front of you? Is that what you're doing?

Steve: Yeah. Of course. It's my beloved HP. Oh, I hit - I meant to swap...

Leo: Have you tried - completely off purpose. Go ahead and keep typing. Have you tried any of the iPad calculators? There's one called "42" that's really got...

Steve: That's the one I have.

Leo: It's awesome.

Steve: I love 42. Of course we know why it's called "42."

Leo: We do. It's the answer to life, the universe, and everything.

Steve: Okay. So here we have a 10-character password using only upper and lowercase and the digits, with 5.94 binary bits of strength per character, gives us 7.6 x 10^17 possibilities.

Leo: That's a lot [laughing].

Steve: 7.6 x 10^17.

Leo: I think that's the number of stars in the galaxy or something like that. It's a large number.

Steve: That is a - yes. And this is going to - they're randomly chosen. They're gibberish. I would say no reason to go any stronger. You can if you want. But someday you might find yourself, for whatever reason, needing to type one of these in manually. For example, the - oh, did I mention that all this is free?

Leo: There is a paid version. I mentioned at the beginning I pay a buck a month. But everything you've said so far, except for maybe the database dump...

Steve: No. It's the mobile stuff.

Leo: The mobile stuff's not free.

Steve: The mobile browser stuff is what they charge for. They have said that they're going to reserve the right to put ads somewhere, like some of their stuff is web-based, where you end up using a secure web page, for example, to view your vault. You're able to view it locally also. The browser plug-ins open it up. In fact, in Firefox it's chrome:// and then the path to the file, which they provide. So...

Leo: By the way, 10^17, it's the number of meters in a light year.

Steve: No kidding.

Leo: I'm just looking up orders of magnitude.

Steve: Very cool.

Leo: And it is roughly - it's a little less than the number of stars in the galactic arm.

Steve: That ought to do it, yeah. 7.6 x 10^17. Good luck guessing that, FBI. Anyway. So...

Leo: I'm sorry. We were talking about the free stuff, what's free, what's paid. And the mobile is - yeah.

Steve: So the stuff you pay for. So conceivably they could put some ads somewhere. I haven't run across any yet. And they said they will be tasteful little Google text-style ads. I mean, these seem to be great people from everything I've seen. And I've got some of the questions that I had answered through some email exchange, and they've been also very responsive. So it's the mobile stuff that you pay for. So the mobile versions of the applets. But all of the browser-based stuff, all of the regular browser, I mean, like, phone mobile is what I meant, is free, as is the iPad tabbed system is free. Let's see. Premium, oh, they do have - we'll be talking about one-time passwords in a second.

Leo: And YubiKey support and all of that stuff.

Steve: Yes, yes, yes, yes.

Leo: But you know the real reason I pay them a buck a month? Because it's cheap, and it supports the further development of the program.

Steve: Yes, these guys deserve it.

Leo: I don't pay them for the additional features. I pay them because they deserve it.

Steve: Yes, exactly. So my point was that I have the LastPass Blackberry app on my Blackberry. My Blackberry now contains an encrypted copy of my master database. And if I ever need to log in somewhere where I have nothing with me but my phone, well, first of all, anything I'm logging into is a browser. So any browser that I encountered, I

could use a bookmarklet in order to log in.

Leo: That's how I do it on the iPad, by the way. And the iPhone. That really works well.

Steve: Yeah. It just works perfectly. It fills in the form for you, it's like, oh, well, off you go.

Leo: It's kinda neat.

Steve: It's very cool. It's like magic, Leo.

Leo: It's hard to believe.

Steve: It's like I'm so spoiled now.

Leo: Don't tell Steve it's using JavaScript. Shhh.

Steve: Yeah, we'll be talking about that in a minute actually.

Leo: Shhh. Okay.

Steve: So I would have my phone with me. So there I have the ability to authenticate to my phone locally, that is, using, without any connection, I give my phone my LastPass username, which is my email address, and my LastPass password. That unlocks in there the copy that my Blackberry has. And I can look up my username and password for any of the sites in that database and unlock my own private note vault of stuff that I want to, like, for copying, pasting, or whatever, because they also support this idea of notes.

And remember they do form fill-in. I have had it successfully now populate several different ecommerce sites that I went to with my credit card information. I know LastPass only has it in encrypted form. And this decryption happens on the fly, in the browser, using JavaScript, or where there's a plug-in present, in code in the plug-in. So it's secure. So because it's conceivable that I might need to manually type the password in, 10 characters, which we've seen is 7.6 x 10^17 possible combinations, that's feasible for me to read off the screen of my iPhone or my Blackberry and manually type in, uppercase A, lowercase z, 0, 2, uppercase Q, lower, you know, and so forth, and be able to log in. So that's covered.

However, we want to make sure that we - so what we've done is we've concentrated a huge amount of value into our LastPass authentication because authenticating with LastPass now literally unlocks the keys to our kingdom. So how is that made safe? Well, it's made safe through several techniques that we've talked about in the podcast in the past. They support something very much like my own Perfect Paper Passwords. They call it the Grid.

And using the - when you log into LastPass and authenticate, you can have them generate for you a random grid, which is A through Z and maybe 0 through 9. I think that's all it is. And it's very much - they liken it to Battleship because - so it comes up on the page on your browser. You print it. You snip it out, and you carry it in your wallet. When you activate the grid, then any authentication is challenged by, after you give them your username and password, they challenge you by saying tell us the characters at B4, C9, Z7, and Q2. And so you've got to get out your grid; and, like playing Battleship, you type in the character at each of those coordinates...

Leo: That's cool.

Steve: ...in order to say this is really me. Because as we know, this is another factor of authentication. Now it's something I have in addition to something I know. I know my LastPass password. Now it's also something that I have.

Leo: Is that kind of how Perfect Paper Passwords is almost like - no.

Steve: Well, Perfect...

Leo: It's not a matrix.

Steve: The one thing that Perfect Paper Passwords does is it never reuses any of the strings.

Leo: It's the one-time thing, yeah.

Steve: Yes, it's one time. Now, the grid could potentially, I mean, and your farthest worry would be that something is watching you log on using the grid and is learning your grid. So the only thing I would - and I don't know, I meant to ask, but I forgot to ask, if they have any, like, you need to update your grid. They absolutely allow you to kill your grid and make a new one. So, for example, if you lost it, or if you left it somewhere, you'd want to kill that one and make yourself a new one, which would be completely re-randomized and brand new. And the other one would die and no longer be useful. Oh, and that's true of bookmarklets, by the way, also. You're able to deliberately kill any bookmarklets that you might have left on a different browser somewhere, or you might have temporarily installed somewhere. If you change your LastPass password...

Leo: That happens anyway because I've done that.

Steve: Yes.

Leo: And none of my bookmarklets work anymore.

Steve: Yes. If you change your LastPass password, then - and you can tell, Leo, I really learned this thing.

Leo: Yeah, because, I mean, I've been using this for a year, and you found stuff I hadn't even discovered yet. So, yeah.

Steve: So if you change your LastPass password, that obsoletes your bookmarks. But you can also explicitly do that just for security purposes, although you will then have to recreate them manually. But that's easy because you're just able to drag and drop them from the page onto the shortcuts bar. So I would say, if you're a grid user, and you're a heavy grid user, just obsolete the grid yourself. I don't know whether these guys do or not. Obsolete the grid yourself every few months so that, I mean, it's very far-fetched that anything is memorizing your grid behind your back, but that's the only possible problem that I could see where Perfect Paper Passwords is a little bit better.

So the other thing they support is a very cool software one-time password generator. Oh, and you can even make one-time passwords. That is, using their web interface you're able to say, generate for me some one-time passwords. And you can click on it a few times. I made up 10. And they're long mothers. They're, like, whoa, okay, I'm going to print this out or write this down very carefully. And you can use those if you know in advance that you're going to be somewhere where you want to make sure no one can ever log in again.

And so they just, like, emit some - every time you ask for one, they'll generate another one. And you can print them out and carry them with you and then just, like, cross them off as you use them. So and they're good until you kill them. And you're able to kill them at any time. Or you use them exactly one time. So they have a built-in one-time password-generating facility there.

They also have sort of the equivalent of a software YubiKey. They do support our very favorite Stina Ehrensvrd Yubico's YubiKey. And I tried it out, works great. You just go there, and in their configuration dialogue on their website, they have got a tab, a YubiKey tab. You go there. It's got room for I think maybe eight or maybe 10 different YubiKeys, so you're not just stuck with just one.

Leo: Oh, that's good because I worry about losing any kind of a dongle.

Steve: Right. And so you're able - so what I did was I just, I plugged it in, set the cursor there, touched the YubiKey, it saw, it got my YubiKey string, and that's all I had to give it because remember the YubiKey has an ID as part of it, and then the one-time password portion. And so you look up the YubiKey. They checked with Yubico Central and said, yup, we know about that YubiKey, and now we're ready to go. So you're able to have multiple YubiKeys.

And so what that would do is that's used anytime you need to authenticate with LastPass. And you can choose the security level that, like, with multiple checkboxes of, like, I only want to - I want to be able to generate passwords with LastPass, that is, have LastPass do all of its things, and only authenticate once. I want to require authentication every time. I want to authenticate after so many minutes. So there's various ways you configure lockouts and the need to reauthenticate.

They do have one of their premium features, which is the $1 a month thing that Leo was talking about. They have something called Sesame which - as in Open Sesame. It's kind of cute. It's an app, again cross-platform - Windows, Mac, and Linux - which is - it's like a software one-time password. So it does the same thing that the cool little YubiKey does...

Leo: Oh, that's what I want.

Steve: Yes.

Leo: Because then I can't lose it.

Steve: Exactly. Exactly. So you download it from them. You stick it on the USB...

Leo: Any USB key.

Steve: Any USB, as many as you want.

Leo: I like that.

Steve: You need to authenticate to it. But when you do, it'll automatically log you in on your - it'll launch your browser, log you in, and authenticate you, all in one process.

Leo: I'm getting it right now.

Steve: It is very cool. And they do have support for IE Anywhere, which is the sort of the mobile transportable version of IE, which is useful because of course IE is pretty much everywhere. And the last thing is import. They do allow you to import from pretty much everything I've ever seen. And I'm going to run through it just once. So if you're currently somewhere else, and I've sold you, as I have sold myself, on this thing being, like, the answer, you can import from Firefox's Password Manager. I had been using Firefox's Password Manager. Instantly, LastPass knew about everything that Firefox knew, which was extremely cool for me. Also from 1Password; from Clipperz; from something called Darn! Passwords!; from eWallet; from FireForm; from HP Password Safe; from KeePass; obviously from LastPass, it's able to import its own file, by the way; from MSI PasswordKeeper; from MyPasswordSafe; from Passpack; from Password Agent; Password Corral; Password Dragon; Password Keeper; Password Safe; Passwords Max; from PINs Password Manager, from RoboForm, from SplashID, from Sticky Password; from Sxipper, I guess; from TurboPasswords; and from a Generic CSV File. So pretty much everybody there is, you can instantly suck your database in using their import feature of the plug-in, and you're moved over to them.

Leo: It's amazing.

Steve: They just did it. Oh, and form filling is very cool, too. You can give it multiple credit cards.

Leo: I do that, yeah.

Steve: You can tell it all about yourself.

Leo: These are features that many browsers have. And the point is this is the more secure way to do that. And it turns off those browser features, by the way, if you ask it to.

Steve: And cross-platform.

Leo: Right.

Steve: I mean, it's nice that a browser has it. But then you go to a browser you've never been to. I mean, even just - it happened to me in this last week. I set up a new system, and I said, oh, wait a minute, I have LastPass. So I added the plug-in. Instantly that new system knew everything about my world.

Leo: You know what you've done, Steve, is I've been using LastPass in kind of a fundamental, basic way all this time, very happily. I've used the bookmarks and various features. But now I'm going to go back and turn on some of these advanced features. I love this idea of being able to put a USB key with this application on it. I'm going to start doing that. Because I like multifactor, you've taught me to love multifactor authentication. I'm going to start turning that on. That's a nice thing.

Steve: Well, it is the case that we are now storing all of our eggs in one basket. So you want it to be a safe basket, and you want it to be a basket you can back up, and a basket that nobody else can get your eggs out of.

Leo: [Laughing]

Steve: And they really have nailed it. I mean, I don't see a single problem with this. The crypto is clear and simple. And they've arranged so that they're never going to be in a position of anyone being able to, like, steal their stuff. Notice also...

Leo: Because they don't have it.

Steve: Right. Notice that no subpoena that they're served can force them to divulge your information.

Leo: Right.

Steve: They don't have anything.

Leo: They don't know.

Steve: They have only the result of the best encryption that our world, that AES-256, the best, strongest encryption that we know how to produce, they only have the product of that. They have the encrypted blob, and they have no key to it. They never get the key. It all stays local.

Leo: That's very clever that they did it that way. I think that's incredibly clever.

Steve: Yeah. It's, well, it's correct. They did it right. I mean, from start to finish. And multiplatform. So they're not biased towards Windows and against Linux folks. Windows, Mac, and Linux, across the board, for all for this. It's done.

Leo: Steve, we have come to the end of this show, but not to the end of your Security Now! experience. Go to GRC.com for transcripts. We are going to be late with those again because of my fault, some technical issues we had in the studio. We're a day late...

Steve: Elaine will get them done as soon as she can.

Leo: But they'll be there. GRC.com/securitynow. Also 16KB versions, show notes. You can also subscribe to the show at TWiT.tv/sn for the high-quality audio and video. We also encourage you to watch live when we do it normally, Wednesdays at 11:00 a.m. Pacific, 2:00 p.m. Eastern Time, 1800 UTC at live.twit.tv. GRC.com is also the place to go for SpinRite. Don't forget. Steve, thanks so much for your patience. I'm glad we could do this on a Friday. And we'll see you next Wednesday on Security Now!.

Steve: Well, thanks for coming in for it, Leo, and we'll talk to you next week.

Leo: Take care. Bye bye.