SECURITY NOW!

Transcript of Episode #904

Leaving LastPass

Description: This week, since a single topic dominated the security industry and by far the majority of my Twitter feed and DMs, after a brief update on my SpinRite progress we're going to spend the entire podcast looking at a single topic: LastPass.

High quality (64 kbps) mp3 audio file URL: http://media.GRC.com/sn/SN-904.mp3
Quarter size (16 kbps) mp3 audio file URL: http://media.GRC.com/sn/sn-904-lq.mp3

SHOW TEASE: It's time for Security Now!. Steve Gibson is here, and the topic of the day, really the topic of the whole show, is the topic everybody has wanted Steve to comment on since the news broke late last year about the LastPass breach. What happened, what does Steve think, and what's he going to do? That's coming up next.

Leo Laporte: This is Security Now! with Steve Gibson, Episode 904, recorded Tuesday, January 3rd, 2023: Leaving LastPass.

It's time once again, first show of the new year for Security Now!. Here he is, well rested, relaxed, hasn't had a security problem in weeks - not so - Steve Gibson. Hi, Steve.

Steve Gibson: Yo, Leo. Great to be with you for this first podcast of 2023.

Leo: Wow. Did you have a good holiday?

Steve: I did.

Leo: Good.

Steve: I got a lot of work done, which is for me that's a good holiday. And I apologize to my lovely wife, who just said, "You're a machine."

Leo: Yeah.

Steve: And I said, well, you know, when you love what you do, you do what you love.

Leo: You kind of have to, though, to be a good coder. You kind of keep - you get rusty quickly, so you have to keep going; right? That's been my experience.

Steve: On CNN, Fareed Zakaria had a series of interviews of, like, well, people we all know. And the one that I thought was interesting was Elton John.

Leo: Yeah.

Steve: And, you know, you hear about these guys who are, like, just amazing. But when you look at the back story, like in his 20s and 30s he didn't think he was going to succeed.

Leo: No. Little Reginald Dwight, the Beatles used to mock him. But he knew the Beatles, on the bright side.

Steve: Yeah. Anyway, so I guess my point is that, yeah, you know, I've been programming computers since I was 14, and because I love it.

Leo: Yeah.

Steve: And I have a plan that's going to keep me coding until I'm finally, like, okay, where did the ENTER key go? What? Where?

Leo: Has it gotten harder or easier for you as time goes by?

Steve: I suspect that - I guess it would be a change in characteristics. Certainly the way I approach solving problems in code has matured so that things are easier because I'm not painting myself into corners. When surprises occur, it's like, oh, look. Oh, in fact I had that happen a couple weeks ago because I rewrote SpinRite's logging system completely. Well, it turns out what I had written was pretty good. And so it allowed me, you know, it didn't fight back very much when I was like trying to change the way it worked. And it's like, wow, this was a pleasant experience. I didn't think I was going to have that happen. So, you know, but when you're starting out, you can sort of force the computer to do what you want, as opposed to it just sort of gracefully agreeing. And that's just sort of a consequence of the way you approach code.

Leo: Somehow I think you get it in your mind, and you're speaking a language that is second nature. And so now it's a question of just conversing; right?

Steve: Yeah. And I think in the beginning I was - because I remember I was editing a series of books in my late teens called "Teach Yourself Basic."

Leo: Oh, yeah.

Steve: Bob Albrecht was the author.

Leo: You were 19 and editing Bob Albrecht's books?

Steve: Yeah.

Leo: Wow. Wow.

Steve: Anyway, but I remember coding back then, because I didn't have a lot of experience, when the program didn't do something, you'd go, oh, and you'd like stick in a go-to; right?

Leo: Right.

Steve: And you're like, ooh, I need an if statement here to make it go over here.

Leo: Yeah, a Band-Aid over it, yeah, yeah.

Steve: It was like, it was just like, oh, not right.

Leo: And it got worse and worse and worse.

Steve: Exactly, because you end up just chasing yourself around. Oh, wait. Now it needs to be this, unless it's this, in which case we've got to go over here. So anyway.

Leo: You don't do that anymore. You know, because you have enough experience now. You kind of know the consequence of that kind of behavior.

Steve: Yeah. And I will tell you, though, the same problems or the same issues always exist, which are those off-by-one problems. That's like something fundamental to computers.

Leo: It's universal, yeah.

Steve: It just, you know, do I mean greater than, or do I mean greater than or equal to? And it's like, oh. And so now, you know, I'm just - so I think what you learn is to be very careful about those. You're not going to get rid of them because, you know, you can't. But so, but what you can do is appreciate that there's a landmine every time you're checking to see whether, wait, do I mean that the carry bit is on, or the carry and the zero flag are on?

Leo: Yup, yup.

Steve: So, you know.

Leo: Yup, yup, I totally understand it, yup.

Steve: Okay. So.

Leo: Let's get to the show. Here we go.

Steve: Our listeners will be glad, those who have already left, to know that the title of today's podcast is "Leaving LastPass."

Leo: Yikes. I have to say I left almost two years ago, well, maybe more, for Bitwarden. And then about a year ago I finally said, "I'm going to delete my LastPass vault." Which as it turns out I'm thanking goodness for. I do hope they didn't back it up somewhere, and that that wasn't the backup that was downloaded.

Steve: Yeah.

Leo: There's no assurance on that.

Steve: So, you know, this past couple weeks a single topic has dominated the security industry, and by far the majority of my Twitter feed and DMs, you know, it just exploded. So today, after a brief update on how my holidays went with SpinRite, we're going to spend the entire podcast looking at a single topic, which is LastPass. And I think everyone is going to appreciate this because, you know, there's no hyperbole. There's no hyperventilation. There was plenty of that on the Internet, and we'll talk about some of that, because that's part of what fueled everyone's frenzy. So I'm going to do a little bit of taking down of the takedowns. So the question is, what happened? What does it mean? How worried should you be? What are the consequences? And what's next? And of course we have a great Picture of the Week.

Leo: Very important subjects. And we will get to those in just a second. Picture of the Week time, Mr. G.?

Steve: So this is another one of those. I looked at the picture, and I gave it the caption "There are no words."

Leo: And I immediately burst out laughing when I saw it.

Steve: Yeah. I looked carefully at the road behind where the arm is to see if maybe somebody had photoshopped it to like exclude the arm from the picture because how else do you explain this? Okay. So for those who are not seeing the photo or don't have the show notes, we've got your automotive gate, you know, with an arm that comes down in order to block a car from passing. And in fact you can see the sensor loops that are in the asphalt that have the wires going over to the little motor stand that moves the arm up and down. So it all looks real. The only problem is the arm extends about, like, a foot into the roadway.

Leo: Yeah.

Steve: And it's a two-lane road.

Leo: There's a lot of room to drive around the arm, let's put it that way.

Steve: Yeah, I just, you know, somebody went to a lot of trouble. This is a bright yellow pedestal with a blue-and-red striped, like visible arm. You know, I was going to say so you don't hit it, but it'd be hard to hit it because it's like not in the road.

Leo: Yeah, that's a good point. It's not even like you could hit it. You'd really have to work at it, yeah.

Steve: Maybe if your passenger door was open when you were driving by.

Leo: It's just a suggestion. It's a security suggestion.

Steve: Oh, wow. Anyway, thank you whomever sent this. I appreciate it. Okay. As I said, if there was any security news that happened since our last podcast, I never had time to track it down since I was digging deep into today's topic, which exploded my Twitter feed. So next week I'll be looking back over everything that has transpired in the security world - other than LastPass - since our last podcast.

Leo: Oh, good. Oh, good, okay.

Steve: And we'll catch up. So, you know, nothing is getting by me. Next week we'll be brought current from the last time we talked. Also the last time we talked I mentioned that the eighth alpha release of SpinRite had been put out on the prior weekend before the podcast. There has been no release since. We're still on eight. And the testers are getting anxious. They're like, hey, we were having a lot of fun here. What happened?

Okay. So mostly those releases were intended to resolve a couple of mysterious behaviors that people were reporting on seriously damaged drives when they were testing SpinRite against what it turns out they have literally - our testers have boxes of old hard drives that they've been holding onto in anticipation of this day. So I've got pictures of them, like boxes. And they're like, they have scoreboards of SpinRite has recovered these 11, and it's got problems with these four, and we're not sure about a couple others. It's sort of - it's a lot of fun.

Anyway, because I was focused on resolving those few mysteries, it was just difficult for me conceptually to move on because there were some problems that I didn't understand. So I wasn't fixing the growing number of non-mysterious things that people were noting. Those were things like during intense data recovery SpinRite was not updating its onscreen clocks. They were just froze while SpinRite was like grinding away on a hard drive, trying to recover a damaged sector. Or if the system hung or crashed, the permanent log of all the work that had been done up to that point would be lost because SpinRite's log was deliberately being kept in RAM until SpinRite's graceful exit. But unfortunately, if there was not a graceful exit, you'd get no log. So anyway, thanks to all of the feedback during that first flurry of testing, I saw that SpinRite could be much more aware of drive trouble now that it was no longer isolated from the drive by the system's BIOS, and more.

Essentially, the feedback from that extensive initial testing showed me some ways in which I could improve upon my whole first take. You know, I want 6.1 to be as good as it can possibly be, not only because we're going to be living with it for a while during the work on SpinRite 7, but because every way in which I can make SpinRite 6.1 better today will be inherited by SpinRite's future. So none of this work is going to get tossed.

So I've reworked some significant portions of SpinRite over the holidays. Logs are now being incrementally written to non-volatile media so that everything that has happened so far will be saved in the event of a system hang or crash. And if the user wishes to write SpinRite's log back to the same drive SpinRite's running on, then it will be spooled to RAM and written at the end, and they're notified that's going to happen so they could change their mind if they want to. I've also completely rewritten SpinRite's clock management and completion estimation system. It's now also providing much more feedback about the state of critically ill drives, which it turns out we have in abundance, thanks to the testers that we have pounding on SpinRite.

So I'm left now with a bunch of to-do list items. Our GitLab has a lot of feedback. So I'm going to continue working on this ninth alpha release, that is, when I release it that's what it'll be, because my goal is to basically finish it again, get everything done that I know of that can be finished so that people are not reporting things that I already know I need to do. I'm going to do all those first. And then release nine, this alpha release nine, ought to be like as close to beta as it can be. I'm sure there'll be a few more iterations. I suspect that these really weird drives are still going to give SpinRite some heartburn. But, you know, now's the time to take an antacid.

So, okay. We last talked about LastPass following the November 30th disclosure of the second breach. So that was, you know, in each of these breaches, the one back in August and now this one in November, there have been a pair of disclosures. There's the immediate acknowledgment that something happened, and then about three weeks later we get a more fulsome explanation of exactly what that was that happened. After the first announcement in August I noted that their follow-up occurred exactly three weeks, 21 days later, with the results of that forensic analysis.

So when we talked about this on November 30th, our listeners may remember that I noted that we might expect to receive an update on the second breach three weeks after that. Three weeks after Wednesday, November 30th was Wednesday, December 21st, and the follow-up arrived one day after that on December 22nd.

Now, the "likes" and the clickbait-driven tech press and some security researchers' postings suggested that LastPass had deliberately timed their bad news for release shortly before Christmas in some attempt to have it swept under the rug. Given the timeline we've seen LastPass adhere to previously, I think that's not the case. And, you know, it's the kind of junk that causes any careful reader when they see that, if they understand that that is actually where the timing came from, to wonder whether what follows will be objective reporting of facts or a subjective smear.

Two days after Christmas, a researcher named Jeremi Gosney, who is a widely recognized expert in password cracking, wrote a lengthy and inflammatory takedown of LastPass over on his new Mastodon account.

Leo: Yes, I read that, yes.

Steve: Yeah. He recently moved there from Twitter. And in fact over on Twitter he said: "I won't be posting anything new here. I'm over on Mastodon." So while Jeremi made some excellent points that I will be touching on here in a few minutes, he also suffered from the "piling on" syndrome. You know, for example, he wrote, quoting him: "LastPass's claim of 'zero knowledge' [he has in quotes] is a bald-faced lie. Nearly everything in your LastPass vault is unencrypted." He says: "I think most people envision their vault as a sort of encrypted database where the entire file is protected. But no. With LastPass, your vault is a plaintext file, and only a few select fields are encrypted."

Okay, well, you know, what does one say to that? There's nothing about that that accurately conveys the truth, or even a sense of the truth. As evidence of that, another well-known researcher, Wladimir Palant, the creator of Adblock Plus, who blogged a lot - he blogs from his site titled "Almost Secure," and he had four blogs over the holidays about this incident. The day before Christmas, Wladimir posted an article titled "What data does LastPass encrypt?" And - this is important - he provided a snippet of JavaScript code that could be dropped into any web browser's developer console, which we all have, to retrieve the logged-on user's data blob, you know, their vault, directly from LastPass's cloud server.

And I'm going to suggest that everybody listening to this podcast do this because there are some fun things we have in store. In the show notes I have this little three-line snippet of code. So any current LastPass user can obtain their encrypted vault data to examine it for themselves. Open any browser. I used Chrome for this, and I also tested on Edge. Log into LastPass, assuming you still have an account there, so that you're looking at your vault page, which is what you get when you log into LastPass. Press F12 to open the Developer Tools. I think there's also some other way of getting there, probably under the main menu. I'm sure you can go to Developer Tools. F12 is what I use. And your screen will basically split into your web page on the left and this whole bunch of stuff maybe you've never even seen before on the right, which is where all of the magic under the browser happens.

Select the Console tab to move to the Console view in the Developer Tools, and you'll have a cursor. So paste the short three-line JavaScript query which is shown above in the show notes into the Console and press ENTER. If everything worked, the screen there will fill with a large XML format dump containing name-value pairs. But the vault is far larger than your page. So look carefully at the bottom of the page where Chrome or Edge will be saying Show More, where you can like ask it for another chunk, but also offering a Copy button. Click Copy, which will move all of that query response data onto your machine's clipboard.

Open a text editor - I use Notepad++ on Windows - then paste the clipboard into the editor and save the file so you don't lose it. Okay, now, there's a bunch of stuff there that you can peruse. Passwords begin with p, with a letter p, lowercase p, and an equal sign, followed by a double-quoted string. AES-CBC encoding has two parts, the first being - I'll be getting into a little bit more of this here in a minute, but I just wanted to go over this part first - has two parts, the first being a pseudorandom initialization vector which CBC requires, and the second half or part of it being the encrypted string itself. The older format passwords will only have a single ASCII-encoded string, which is the ECB format encryption of the data since ECB - that's the Electronic Code Book we were actually talking about back when we were talking about Microsoft Office 365 in October. It does not use an initialization vector. And that's the reason its use has been deprecated.

So then with Notepad++, since you're able to do Regex searches there, I wrote a quick Regex to scan my encrypted vault for any non-CBC-encoded or encrypted passwords, and I didn't find any. And I put the little Regex expression also in the show notes for anyone who's interested. Now, that's not conclusive. But I'm pretty sure that my vault contains very old and original passwords which predate LastPass's switch to CBC. And we'll be getting to why that's important here in a second. So anecdotally I'm not seeing evidence of old CBC encryption that was not autonomously upgraded for me. And again, I wanted to make sure that everyone understood that they are able to easily obtain their encrypted LastPass vault blob and have a look at it for themselves. And I'll have more to say about that in a minute.

So anyway, for those who are interested, Wladimir's blog posting is titled "What Data Does Lastpass Encrypt?" And it provides additional details beyond what I've just said, and I've linked to it in the show notes. But on the question of whether any sensitive unencrypted information is contained therein, I guess the question is what is your definition of "sensitive." And that will vary by user. It is certainly the case that the LastPass vault does contain unencrypted information. Presumably this is for some features that they wanted to provide. It's not exactly clear why they're doing this.

We've got, as you were saying before, Leo, a bunch of smart coders that are listening to this podcast. You were talking about the Advent of Code and your work on that over the holidays. I have asked, and I am here asking, for some interested hacker/coders to write a simple utility to deobfuscate all of the non-encrypted data in the LastPass blob so that we can all see what it looks like. I developed SQRL. I could obviously do that myself. I'm going to stay focused on SpinRite because that's where I should be. And the hacker/coders would rather I finish SpinRite...

Leo: Yes, we would.

Steve: ...than get distracted by this anyway.

Leo: Well, how is it obscured?

Steve: It's just BASE64 and HEX encoded.

Leo: That's what I thought. Okay. So that's pretty straightforward.

Steve: So it will be a simple matter to write some code. It could be Perl. It could be Python. It could be whatever. It would be really cool, except for the privacy concerns, to have a website where you could dump the file in, and it would give it back to you unencrypted. That way people, you know, it would just be easier for people to use. But they probably don't want to be handing their LastPass vault, even though it is encrypted, to some random website. So I think an app that you can run on your local machine that deobfuscates this LastPass vault blob so that you can see the URLs, you can see the email addresses, and the things that look like just HEX junk now, well, those are not encrypted. They're just obfuscated in order to send them back and forth across the Internet because, as we know, you're unable to send any special characters across a seven-bit channel.

So I already posted this question and a request for such a utility over in the Security Now! newsgroup on GRC. Everybody listening to this podcast is able to do it. You know, shoot me a tweet or send a note to Greg at our support address at GRC so that I find out about it. And I will post the outcome, the results, tell everybody a week from now on the podcast how they're able to deobfuscate their blob. I think you should go grab it now, though, because LastPass could change this if they decide they don't want everyone grabbing it. On the other hand, they probably have to obsolete all their clients in order to change it, so it may be staying where it is. Anyway, I've got mine. You should probably grab yours. And then a week from now we'll have some utilities, I'm pretty sure, which will allow us to clarify what's there.

For what it's worth, someone who replied to Wladimir's post obtained and examined their encrypted blob and was not concerned. On his page where he shows the responses to what he wrote, this person said: "Thank you for this article. As a longtime LastPass user, the latest news was unsettling. I dumped my data following your instructions and took a very careful look. I found that user ID, passwords, account names, account numbers, and especially the notes" - you know, extra - "were all encrypted." Now, okay. This sounds like this person didn't deobfuscate the file, so they may have just been looking for email addresses and usernames and IDs in cleartext, you know, completely legible ASCII. So they may have the wrong idea.

Anyway, this person said: "I did searches on specific sensitive data and literally inspected every page" - and that's a lot of pages - "and found no sensitive data that was in plaintext. On the other hand, all of the URLs were in hex, which to my mind is not encrypted." Okay, so they knew that. They said: "For me this is a lower priority, as my browsing history is collected by everybody these days, as evidenced by instant ads for something I just started searching for. I was considering changing password managers, but do not plan to at this point. It has prompted me, however, to change my master password, which has two-factor authentication protection also, and to review my most sensitive data and ensure it is as secure as possible - strong passwords and two-factor authentication, et cetera. Thanks again for this article. It was most useful." And again, I'm not sure this person actually understands what was lost. But we are all going to within the next week.

Okay. So at this point there are about five different directions I want to take this because we have so much ground to cover, a little more than you might guess. By the end of this podcast I believe you'll have sufficient information to make an informed decision on your own, if you haven't already. And if you have, you'll have more evidence for that decision. And it won't be the result of a bunch of inflammatory and exaggerated half truths.

Okay. So let's examine what LastPass posted, just the meat of it, on December 22nd which has stirred up this hornet's nest. The main issue is a couple paragraphs. They said: "Based on our investigation to date, we've learned that an unknown threat actor accessed a cloud-based storage environment leveraging information obtained from the incident we previously disclosed in August of 2022. While no customer data was accessed during the initial August 2022 incident, some source code and technical information were stolen from our development environment and used to target another employee, obtaining credentials and keys which were used to access and decrypt some storage volumes within the cloud-based storage service."

They said: "LastPass production services currently operate from on-premises data centers with cloud-based storage used for various purposes such as storing backups and regional data residency requirements. The cloud storage service accessed by the threat actor is physically separate from our production environment. To date, we have determined that once the cloud storage access key and dual storage container decryption keys were obtained, the threat actor copied information from backup that contained basic customer account information and related metadata including company names, end-user names, billing addresses, email addresses, telephone numbers, and the IP addresses from which customers were accessing the LastPass service."

Leo: This is a little bit more extensive than their initial claim that just some customer information was exfiltrated.

Steve: Yes. And then here's the big one. The threat actor was also able to copy a backup of customer vault data from the encrypted storage container which is stored in a proprietary binary format that contains both unencrypted data, such as website URLs, as well as fully encrypted sensitive fields such as website usernames and passwords, secure notes, and form-filled data. Which is exactly what we - all that we've been talking about, this vault blob. So yes, Leo, exactly as you say, this is, you know, this is what - this on December 22nd is what blew the lid off of this is that LastPass confirmed that unknown actors have acquired their customers' vaults.

Leo: All of the vaults? Some of the vaults? They're not very forthcoming in this.

Steve: They're not.

Leo: In fact, I think they're criminally not forthcoming, to be honest. Every single customer should have by now received an email saying yours was one of the vaults. Have you received such an email? Has anybody? No.

Steve: Well, we all received the email of this.

Leo: This is generic, though. Is it mine? Is it your vault? Is it his vault? Whose vaults? All of them?

Steve: So one presumes, if it was not all, they would have said "some" because that would have been better than not saying anything. No COO worth his salt is going to say, "Oh, yeah, they got everything. They got the whole farm. In fact, we haven't seen our cleaning people." You know, who knows?

Leo: Yeah.

Steve: Okay. So anyway, that last...

Leo: So when they say "a backup of customer vault data," it's of every bit of that data.

Steve: Yes, yes, yes.

Leo: You can assume that.

Steve: There's no other way to read that. If they would have said "a fraction of our customers" or something like that.

Leo: Some, yeah, yeah, okay, okay.

Steve: So LastPass admitted that their offsite cloud backup of their customers' mostly encrypted, but also some not encrypted, vault data was exfiltrated and is now in the hands of malicious actors. This is the eventuality that all of this TNO, you know, Trust No One, client-side, encrypted before it leaves your computer, technology was designed to address. The idea is that, if the user has chosen a strong and unguessable master password, the only thing the bad guys get, or got, was some less critical metadata, including some web browsing and IP usage history.

Okay. So one thing needs to be made very clear, and I've seen some misunderstanding about this already in Twitter. While changing and strengthening your LastPass master password now, or changing the iteration counts now, which we'll be getting to in a second, would make any next similar loss by LastPass...

Leo: Any future hacks.

Steve: That's right, less problematical.

Leo: But that vault data was encrypted with your old master password.

Steve: Correct.

Leo: We don't, by the way, we don't know how old because we don't know the date of the backups. There are so many missing pieces of information.

Steve: We can at least hope they were being conscientious about backing up.

Leo: But was this the main backup? Or just some...

Steve: We want the bad guys to have the latest and greatest.

Leo: Got to have the latest.

Steve: Like, yeah, that's right, we want fresh backup data.

Leo: But you understand my question. You can't assume that it's your current master password. What if you had monkey123 some years ago.

Steve: Right.

Leo: You don't know. Was that in the backup? What vault did they back up?

Steve: Good point. I think that the lawsuits will probably clarify a lot of this information.

Leo: Yeah.

Steve: I think that's where we're going to be finding out. So anyway, just to finish, the vault blobs, as you said, that were stolen were encrypted under the LastPass master password at the time of that backup. We don't know what that was, but it was before today.

Leo: Right, obviously.

Steve: Whenever you just changed it back, you know, after this announcement. So we want to be clear about that. Okay. So let's talk about cracking these mostly encrypted vault blobs. Way back before some clever hacker realized that they could program a GPU to run a cryptographically strong hash function very, very quickly, passwords were protected by hashing them, like, once. You know, that was plenty since the crypto geniuses who designed these hash functions did so specifically to provide that guarantee, which was that passing a password through a hash once would produce a unique value that could not be reversed to obtain the password that was originally fed into the hash. Okay, so no problem; right? Well, not quite right.

The way to defeat this mechanism is to get a super-fast hashing engine, like a GPU, then run every possible password through the hash and see whether what comes out matches what came out when the user originally hashed their password. So somewhere along the way hackers realized that doing that every time they wanted to crack a password was highly redundant. And this was around the same time that hard drive mass storage prices were falling. So the concept of a "Rainbow Table" was born.

The idea was to run all possible passwords through a common hash function that all the websites were using at the time, and do it just one last time, run every possible password through the hash function one last time, but this time store the result along with the password that was input. Then, when presented with a hashed password that had leaked from some website breach, you just do a lookup in this rainbow table. You'd look for the same hash, and then that would tell you what password was used to get that hash, and then you'd go run around to the sites that that user uses and log in because everyone was only using one password back then also.

Okay. So the trouble was, hash functions were designed to be both secure and efficient where efficient means fast. But fast is the reverse of what we want for passwords, since we want to prevent this sort of brute-force password guessing to obtain a hash. Okay. Therefore Password-Based Key Derivation Functions, generically known as PBKDF, and there is a popular one that is actually called PBKDF2, they were created. There's a handful of popular PBKDFs that have various features and tradeoffs. But most rely upon iterating some core function, doing something some number of times.

When I was working with SQRL, I wanted something that was GPU-resistant. And so I used Scrypt, which is a so-called memory-hard function, because there's no way to short-circuit the need to have a chunk of memory that is filled up by the password, creating pointers that the algorithm has to jump around between within this memory.

The reason this is a GPU defeater is that GPUs have lots of computational power, but typically not lots of memory per core. So that's an example of a PBKDF that is different than iteratively hashing the SHA-256 function a whole bunch of times in order to slow things down.

But anyway, so that's what these things are. As I said, most of them rely on iterating some core function. If performing a single password hash is too fast, and it is nowadays, then you hash the hash, and then hash the hash of that, and then hash it again and so forth. Okay. So where I'm headed with this is that back in 2008 when Joe Siegrist founded LastPass, a single hash iteration provided sufficient security in 2008. GPUs had not yet been developed and deployed for hashing, and computers were much slower than they are today. You know, the cloud didn't exist where you could just borrow some insane computational resources for a few milliseconds. But through the years, all of that has changed.

So LastPass began iterating their PBKDF. The switch from one iteration, the original one iteration to 500 iterations, thus making it 500 times stronger, on the other hand it was 500 times very weak, occurred in June of 2012. Okay. So that was four years after LastPass's initial launch. That only held for about eight months until February of 2013, I guess more like six months. Anyway, no, about eight, yeah. February of 2013. That's when LastPass's default iteration count was jumped by a factor of 10 from 500 to 5,000. And then finally, five years after that, nearly five years ago in February of 2018, the last jump was made from 5,000 to a heady 100,100, that is, one zero zero one zero zero iterations.

Leo: And he told us about this when they did it, and told us to up it; right?

Steve: Yes. If this sounds somewhat familiar, right, it's because we talked about this change on the podcast at the time, five years ago. Okay. So the strength of the PBKDF2 that LastPass uses has moved forward, and the last update was five years ago, bringing it to a little over - 100 more than a 100,000 iterations. Okay. So let's switch now from password hashing iterations, we'll be coming back to that in a second, to the strength of the password itself.

LastPass originally enforced a minimum master password length of eight characters. Again, 2008, okay. But that, too, needed upgrading as our systems became faster, and as asking users to invent stronger passwords became more socially acceptable. Like, you know, monkey123, no, I don't, you know. Hopefully no one's using that anymore. And remember how far we've come from the days when a user's single password that they used across all their websites, it's like, "That's my password," was written on a yellow Post-it note stuck to their large CRT screen border. You know, that's - or maybe under the keyboard, if it happened to slip their mind. You know, back then passwords were regarded as a sheer annoyance, with no perceived value. Look how things have changed.

Five years ago, in 2018, LastPass decided that they needed to update their minimum eight-character password length and to recommend using numbers and both uppercase and lowercase letters. But Wladimir Palant noted something distressing about LastPass's master password strength policy. He wrote: "When LastPass introduced their new password complexity requirements in 2018, they failed to enforce them for existing accounts."

Leo: Mm-hmm.

Steve: In other words, LastPass chose to leave their existing customers' possibly too short eight-character passwords alone. Maybe they didn't want to ruffle feathers. Or perhaps they were worried about the support requirements of, like, people calling up and saying, "Wait a minute, what? I have to change my password?" And presumably, anyone who has changed their LastPass password in the last five years will have been required to strengthen it in the process when setting up its replacement.

While it's true that LastPass has no idea about the length of their customers' passwords, since all they receive is a hashed blob, it would have been trivial for LastPass to add some logic to the LastPass client code to scold their users when the client notices that the master password being submitted no longer meets contemporary complexity requirements. Most users would have taken heed of such advice from their password managers. And again, these days we've all grown accustomed to such requirements.

So assuming that Wladimir is correct, and I have read another anecdotal account from someone who, today, actually it was yesterday, still has an eight-character LastPass master password protecting an unused account, so that suggests this is the case, this means that an unknown number of LastPass user accounts, whose data was recently stolen from the LastPass cloud backup, may have only been protected by an eight-character password.

And unfortunately it gets worse. There are also reports - and I have, since I tweeted about this a couple hours ago, I have confirmation from multiple of our listeners. There are reports within the LastPass-watching security community that LastPass's PBKDF2 iteration count, which was also jumped from its too-low setting of 5,000 up to 100 more than 100,000 in 2018, was not pushed out to all LastPass users.

Leo: Oh, that's very bad.

Steve: And I wrote in the show notes yesterday, if this is true, it's truly horrifying.

Leo: Because that's protecting you against brute force; right?

Steve: Yes. Because taken together it means that there might be LastPass vaults which were allowed to exist for the past five years, since those improvements were made, which are protected by both short eight-character passwords and only hashed 5,000 rather than 100,100 times. And I started that sentence saying, if this is true, now we know it is. I have a bunch of people in my Twitter feed who looked after they saw the show notes which I posted about an hour before the podcast, that made them look in their LastPass vault, and their iteration count was still 5,000.

Leo: So it didn't automatically reset it to a higher number.

Steve: No.

Leo: Now, if you were a listener to this show, you'll remember that we at least had set it to 50,000. I remember you saying turn it up to 50,000. That was before the 100,100 [crosstalk] better.

Steve: And really the only expense is time. That is, the time it takes the hash when you enter your credentials, and then your client says, yeah, okay, you know, like that successfully decrypted the vault blob, so you've entered it correctly.

Leo: It's more important on a phone, for instance, which has a slower CPU than a desktop.

Steve: Yes, yes. But on the other hand...

Leo: But how much time? An hour? No. A few seconds.

Steve: Oh, no, no. It's a few seconds. And that's why, you know, I'd say set it to a million.

Leo: Yes, set it as high as you can; right?

Steve: Yeah, because - and the other thing, too, is - and we'll be talking about this in a bit because there was some controversy that Jeremi raised about how bad LastPass was at keeping secrets. The point is we're also not entering our master password often, unless we've decided we want lots of security, and so it should always be expired.

Leo: And nowadays on most devices, including your phones, you can use biometrics in lieu of reentering the master password.

Steve: Yup.

Leo: Which I think would bypass that PBKDF2; right? I don't know what the...

Steve: So, no.

Leo: That would still do the unlocking bit.

Steve: No. So what it would do is you enter your master password once.

Leo: Yeah.

Steve: Go through that length of time. Then the result is encrypted under biometrics.

Leo: And stored locally.

Steve: And stored locally.

Leo: Okay.

Steve: So then you apply your biometrics.

Leo: So biometrics would be instant.

Steve: And, yes, it ends up being instant.

Leo: Yeah. And that's what I do. Plus somewhat, I would say, marginally more secure; right?

Steve: Significantly more secure.

Leo: Significantly, yeah, okay.

Steve: Yeah, yeah, yeah.

Leo: Provided that your biometric system is effective as it is on Apple.

Steve: Yeah, the way all of these PBKDF2s are designed, the time required is a linear function of iteration count. So a million is a flat 10 times stronger than 100,000.

Leo: Okay. So that's not bad.

Steve: No, that's good.

Leo: Worth doing. All right.

Steve: Okay. So now I want to share some of my feelings about some comments recently made by the well-known password cracker Jeremi Gosney. And since everyone listening to this knows quite well who I am, I want to introduce Jeremi a bit. The relevant paragraph of his LinkedIn bio says: "I am also a core developer of Hashcat, a popular OpenCL-based open source password recovery tool, and I am widely regarded as one of the world's top password crackers. I was named one of the Top 100 Security Experts in 2013. I was also one of the winners of Cloudflare's Heartbleed challenge, and was one of the first to publish a working private key recovery exploit for Heartbleed. My work and research has been featured in hundreds of news articles, and has even been incorporated into university classes and certification courses. I additionally served as a judge on the Experts Panel for the Password Hashing Competition."

So this guy is clearly a serious developer techie who is well able to understand everything that he sees going on in a password manager. And Leo, after we take our second break, we're going to see what he thinks about this.

Leo: I'm just tooting on - because, you know, I don't use that other thing anymore. Just saying, as I hope Steve discusses the LastPass breach in today's episode, he also makes recommendations for current users. The episode will be available at TWiT.tv/sn later today. Or watch us live, and I want to make sure people know that.

Steve: There's more good stuff coming.

Leo: Okay. We're not done yet at https: - I like typing that "s" - //, thanks to you, TWiT.tv. And Google, frankly, who made us do it. Okay, Steve. Let's move along.

Steve: Okay. So back to Jeremi Gosney. As I said, he is a recognized password cracker/hacker hashing guru extraordinaire. So here's how his posting began. He said: "Let me start by saying I used to support LastPass. I recommended it for years and defended it publicly in the media."

Leo: So did we. So did we. So did Steve.

Steve: Of course.

Leo: Right. It was great.

Steve: "If you search Google for 'jeremi gosney + lastpass' you'll find hundreds of articles," he says, "where I've defended and/or pimped LastPass," he said, "(including in Consumer Reports magazine)." He said: "I defended it even in the face of vulnerabilities and breaches because it had superior user experience and still seemed like the best option for the masses, despite its glaring flaws. And it still has a somewhat special place in my heart, being the password manager that actually turned me on to password managers."

Leo: Yes, yes, yes.

Steve: "It set the bar for what I required from a password manager, and for a while it was unrivaled. But things change, and in recent years I found myself unable to defend LastPass. I can't recall if there was a particular straw that broke the camel's back, but I do know that I stopped recommending it in 2017 and fully migrated away from it in 2019." He says then: "Below is an unordered list of the reasons why I lost faith in LastPass."

Now, I'm not going to drag everyone through all of Jeremi's ranting because I don't think it's rational nor fair to LastPass. Recall that even though Jeremi is clearly a highly skilled and qualified technologist, it was he who wrote in this rant that nearly everything in your LastPass vault is unencrypted. And he says that, you know, he said LastPass's claim of zero knowledge is a bald-faced lie. Well, we already know that the intent, the essence of that is not the case. There are things that are not encrypted, and LastPass has always been clear about that. It's made some people uncomfortable, but the bulk of what we have been using LastPass for is protecting our login credentials.

Leo: I do have to say it's less than ideal. And at least Bitwarden, our sponsor, and 1Password do encrypt that metadata, as well.

Steve: Right.

Leo: And the only reason it's potentially problematic is because now somebody has a list of all the sites you have a password for; right?

Steve: Correct.

Leo: Yeah.

Steve: Correct. And actually we'll be talking a little bit about the metadata consequences here in a second.

Leo: Okay.

Steve: So, okay. So anyway, he talks about zero is not zero. But he knows what zero knowledge means. It doesn't mean nothing is there that is unencrypted.

Leo: There's no such thing as perfection.

Steve: And you do need some things unencrypted.

Leo: Right.

Steve: Like the iteration count has to be unencrypted because you need it in order to perform the decryption.

Leo: Right, yeah.

Steve: So, you know. And it would be interesting to know why there are these metadata things that have been left in the clear. Is it just legacy? Is it just because back then once upon a time it didn't matter? Is it because they are actively harvesting that information and selling it? I mean, we don't know.

Leo: Yeah, that's a good question. Do you know if Joe wrote it that way in the first place?

Steve: I reached out to him over LinkedIn. I don't know if he's even still with LastPass itself.

Leo: He's not. I'm going to guess - he hasn't been for years - that he is enjoined not to say anything negative about the company.

Steve: Yes.

Leo: So I would be surprised if he said anything.

Steve: Yes, and he did not.

Leo: Yeah, he sold it outright. The last time I talked with some people at LastPass, which was about two or three years ago, his niece was still working there, and I was very pleased to meet her. But she said Uncle Joe had long left, although last...

Steve: Was that when we did the...

Leo: Yeah, at the RSA.

Steve: We did the Boston...

Leo: No, not Boston, it was RSA in San Francisco.

Steve: Oh, okay.

Leo: Yeah. And which they hosted a party which we attended in 2020, early 2020. But anyway, they were a little closed-lipped about Joe's continued participation, and it's my sense that he had long left, but they didn't want anybody to know.

Steve: Well, in looking back over the timeline, there was the concern when LogMeIn bought LastPass.

Leo: And that's when it started, I think.

Steve: Yes. And he has a couple LastPass blog postings for like a couple years after that acquisition, and then there's nothing more from him.

Leo: Yeah, yeah. That's when he left. And incidentally, LogMeIn has spun off LastPass as a standalone company. So they are now independent of LogMeIn.

Steve: Right, yeah. And as I understand it, they're owned by a...

Leo: Equity capital company, yes.

Steve: A hedge fund group.

Leo: The worst possible owner of anybody.

Steve: Yeah, yeah. Okay. So the biggest concern is another of those legacy issues which LastPass seems to have been reticent to address. And this gets back to that ECB CBC. Recall that a few months back last October, in our podcast 893, we talked about Microsoft's decision to leave their Office 365 message encryption using the Electronic Code Book, the ECB cipher mode. And remember that to illustrate the danger of that we showed that classic Linux Penguin where using Cipher Block Chaining, the good encryption, it encrypted the penguin's image into a pure high-entropy rectangle of noise. You could see nothing there, it was just gray; whereas the use of ECB left a very visible penguin in the image. Okay?

Leo: Oh, boy. Yeah.

Steve: It appears that LastPass was also originally using the clearly less secure ECB mode to encrypt its passwords. Then somewhere along the way they realized that this was no longer the right solution. So they began encrypting any newly saved passwords with the more secure CBC mode. But for some reason they never proactively reencrypted the original, less secure, ECB passwords under CBC. Now, okay, that's the information that the security experts who have looked at vaults are claiming. My own analysis of my vault yesterday showed no evidence of ECB encryption. You're able to tell because of the format of the string following the p= whether it's CBC or ECB. Mine were all CBC. And I'm pretty sure that I've got some old, long-abandoned, like my password for Hamachi probably is in LastPass.

Leo: Yeah. I certainly did, yeah. Why remove it? Yeah.

Steve: And I haven't changed it. I haven't removed it. It's still there. But it was not in ECB. It was in CBC. So my experience is that mine got updated. On the other hand, I don't know if I changed the iteration count or if they did. But we know that a lot of people did not have theirs changed.

Leo: See, this, by the way, I blame LastPass at this point for not - they need to put out this information. What did they do? When did they do it? What didn't they do?

Steve: Yeah.

Leo: Because why are we having to guess?

Steve: Right. And using hot-tempered security experts who...

Leo: And JavaScript pasted in the developer options of their website to figure this out. We shouldn't have to do this. They need to come - they've been, I think, less than forthright.

Steve: Well, you know, I'm sure everything that gets published goes, passes through their attorneys and...

Leo: And now that's what's happening, obviously the same.

Steve: It's amazing that a period came out the other end.

Leo: Yeah. They're doing the minimum disclosure they can legally get away with. Which is disappointing, but I guess I understand.

Steve: Okay. So if this is true, if it is the case that CBC and ECB password encryption is allowed to coexist, for some people they were not updated. As a consequence of that, a user's vault will contain a mixture of old passwords, encrypted under ECB, and newer ones under CBC. I can find no plausible reason for this being left as it has been, and also no verification so far that this is true. Probably a week from now we'll know because I'm sure that a bunch of our listeners are going to be grabbing their LastPass blob vault and run it through the deobfuscators that we're going to come up with. And we'll know if they've got ECB encryption.

But, you know, one thing that's interesting is that if all your passwords were encrypted under ECB, what this would mean is that LastPass on their end, looking at your encrypted passwords, even though they can't tell what they are, they can tell when they're duplicated because that's what ECB does.

Leo: Oh, right.

Steve: Every time the same password is encrypted under the same key, you get the same Electronic Code Book output.

Leo: And we know they can do that because they will warn you you've used this password before.

Steve: Well, but that could be done on the client side.

Leo: Oh, okay. Right, okay.

Steve: So your client who has the vault decrypted, it sees everything.

Leo: That's the difference. Very important to make that distinction between what they can see and what we can see. Yeah, yeah, yeah.

Steve: Right. But what they could see, and maybe this was something that Joe, you know, again, in 2008 you have to pretty much forgive...

Leo: Absolutely, yeah.

Steve: One hash was all anybody was using. And we soon learned, well, let's try 500. Well, 500, we could do that now in a blink.

Leo: I'm convinced Joe did the best possible at the time and continued to keep it up to date as long as he was there. I know you were convinced of that when he showed you everything.

Steve: That's why we chose it.

Leo: That's why we chose it.

Steve: I mean, absolutely. He completely showed me what he was doing. I was able to duplicate and verify the algorithm. It was all open.

Leo: Yup.

Steve: Okay. So anyway, I just wanted to mention that maybe the reason ECB was chosen once upon a time was that, again, from LastPass's view, they could have seen where the same password was being used on multiple sites. They would not have had to rely on the client to tell them about that. And in fact I don't think the duplicate password feature was there in the beginning. That was like a great new feature. It's like, we're going to help you clean up your duplicate passwords and so forth. Right.

Leo: Right, right. Pretty recent. Yeah, pretty recent, yeah, yeah.

Steve: Okay. So that brings me to another of Jeremi's points. He writes: "LastPass has terrible secrets management. Your vault encryption key is always resident in memory and is never wiped. And not only that, but the entire vault is decrypted once and stored entirely in memory."

Okay. And I want to point these things out because people who read his rant got very upset. But some of this is nonsense, and he knows it. As we know, I've been saying recently that it would be nice if the LastPass vault were being incrementally decrypted - I've mentioned that every time we've talked about LastPass recently - so that only the one password needed for login was decrypted from the opaque blob, which after its plaintext was used could be overwritten. But according to Jeremi, that doesn't appear to be the way LastPass manages the user's vault.

And as for the encryption key always being resident in memory, that's a pure requirement of any password manager that is not constantly pestering us to reauthenticate to it. None of us want to be constantly doing that. But if you did, LastPass offers the ability to auto-logout after X minutes of inactivity, at which point it would presumably, and I'm sure it does, wipe all decrypted content from RAM and require the user to login again before its next use. Also remember that when you go to a new site, create a new account, and LastPass says you want me to remember this for you, you click OK. You don't have to reauthenticate. Well, that means that LastPass has added that to the vault, encrypted it using the key which it has to be decrypted for it to do that, and sent the update back to LastPass Central. So again, these things, this terrible secrets management is a requirement for the convenience that we're all getting from every password manager.

So, you know, it's not "terrible secrets management," as Jeremi characterizes it. It's a necessary tradeoff made for convenience, and every other password manager will need to be similarly terrible in order to get its job done without pestering its user to death. And we all need to appreciate that none of the password managers are pretending to protect their users from client-side machine attacks. There is no protection for that, ever, from anyone. That isn't available. We're getting the promise that remote websites cannot access our vaults and that our password manager providers, and anyone who might attack them, also cannot access our vaults. And on that last point it appears that LastPass has made a series of design policy decisions through the years, for reasons only they know, that may have left their users less secure than they could have been in the event of the attacks they have just suffered.

Jeremi also notes that while LastPass's vault key uses AES-256, its 256-bit key is derived from only 128 bits of entropy. If true, that's also unfortunate. Although 128 bits of pure entropy is plenty, why not take the opportunity to generate and use all 256 bits for the AES key? Again, there may be an engineering reason. But overall it feels as though the original security design of LastPass, which in 2008 was ample, with even Jeremi jumping up and down defending it, has not aged well, and that LastPass's caretakers have not been as excited about keeping LastPass on the cutting edge as crypto enthusiasts like Jeremi or I would have been.

And that's why, after having surveyed all of the available commentary, and thinking about everything I've recently learned, I've decided to pull up stakes and leave LastPass.

Leo: Okay. There is an alternative, if you trusted them, but you would have to go through and change all your master passwords. Not your master, all your site passwords, just in case. I know some people are doing that, but that could be a lot of work.

Steve: Well, the problem is, I mean, and that isn't an alternative because what we're seeing is evidence of them not caring.

Leo: Yeah.

Steve: I mean, that's really what this boils down to is them not caring. And anyone who's been listening to this podcast for a while knows what not caring means. Not caring is not neutral. Not caring is the end of life as you know it from a security standpoint because if there is an opportunity to take advantage of a mistake, that opportunity will be taken.

So Jeremi noted that this most recent breach, he said, was LastPass's seventh in the past 10 years. I didn't verify that number, but we all know that they've had their share. And while everyone knows that I'm the first person to forgive a mistake under the theory that they are often unpreventable, this recent breach of their cloud backup provider, which their November disclosure said they shared with their affiliate GoTo, makes one wonder whether some corners may have been cut in the interest of profit. And only insiders know.

But there really isn't any excuse for the engineering decisions they've made which have made the consequences of their now having lost their customer vaults potentially much more serious. They could and should have pushed their legacy users to move to a longer and stronger master password. They didn't. They could and should have had their LastPass clients upgrade all older ECB password encryption to CBC. They didn't. They could and should have absolutely upgraded every user from 5,000 iterations of PDKDF2 to at least 100,100 iterations. They didn't do that either. We now have confirmation of that. They could and should have kept me as a loyal and faithful LastPass user and evangelist. They didn't.

During my post-incident survey of security professionals, three cloud-based password managers kept being mentioned over and over. They were Dashlane, 1Password, and Bitwarden. We all know that Bitwarden is an active sponsor of the TWiT network and a frequent advertiser on this podcast. So I was glad to see other knowledgeable researchers praising it. Here's what Jeremi Gosney wrote once he had calmed down a bit from his being jilted by LastPass.

Leo: I think that's what happened. He was upset, yeah.

Steve: Yeah, clearly. He said: "So why do I recommend Bitwarden and 1Password? It's quite simple," he says. "I personally know the people who architect 1Password, and I can attest that not only are they extremely competent and very talented, but they also actively engage with the password-cracking community and have a deep, *deep,*" he says again with asterisks, "desire to do everything in the most correct manner possible. Do they still get some things wrong? Sure. But they strive for continuous improvement and sincerely care about security. Also, their secret key feature ensures that if anyone does obtain a copy of your vault, they simply cannot access it with the master password alone, making it uncrackable."

Next, he says: "Bitwarden is 100% open source. I have not done a thorough code review, but I have taken a fairly long glance at the code, and I am mostly pleased with what I've seen. I'm less thrilled about it being written in a garbage-collected language" - garbage-collected language - "and there are some tradeoffs that are made there."

Leo: It's C# and .NET.

Steve: Right. "But overall, Bitwarden" - and all you have to do is wipe any plaintext. You know, I did that in SQRL as like, you know, before you release memory, you zero out the plaintext, and then you're fine. Anyway, he said: "But overall, Bitwarden is a solid product. I prefer Bitwarden's user experience, and I've considered crowdfunding a formal audit of Bitwarden, much in the way the Open Crypto Audit Project raised the funds to properly audit TrueCrypt. The community would greatly benefit from this."

Okay, now, I know from my Twitter feed that many of my Twitter followers, or at least those who are tweeting to @SGgrc, are using 1Password. I've not looked closely at it. But from what Jeremi says, that would appear to be a solid choice. Another password-cracking enthusiast by the name of Steve Thomas ranks Dashlane first, Bitwarden second, and 1Password third, but only because Dashlane is using his favorite pet password key derivation function known as Argon2. Argon2 is a memory-hard function designed in 2015 which is highly resistant to GPU attacks.

Leo: It won the hashing challenge as being the most secure.

Steve: Yes. But its implementations need to be careful about side-channel leaks...

Leo: Oh, interesting.

Steve: ...since the original design accesses memory in a password-dependent sequence. Thus you're able to infer something about the password from memory access patterns. As a consequence, improvements have been made in Argon2 since then. But a password manager's choice of its key derivation function is incidental at most, and the strength of any good function can simply be turned up as needed over time. So it appears that all three of these are in the running.

Next, I went to check out the personal plans these three offered. Blessedly, I don't need a family plan or a business plan or anything other than a "Just please keep all of my passwords safe, secure, and synchronized among all of my devices" plan. Although the value of a password manager is now well proven, so that asking for some money should not be a problem, I like the idea of being able to turn people onto it so that they can take it out for a spin without needing to pay in advance for a year's commitment. In other words, a useful free tier, such as LastPass once had but then abandoned, you know, that was part of my criteria in the beginning for LastPass.

Leo: That's the beginning of the end right there, I think.

Steve: And that went away, too.

Leo: Yeah, mm-hmm.

Steve: So that's part of my criteria for the perfect password manager. Unfortunately, Dashlane's free plan only allows for the use of a single device. So you can't use it on both your desktop and even one mobile platform? That's crazy. Why allow any devices if you can't use at least two? So you need to pay at their minimum plan of $33 per year, in advance, which does then provide for an unlimited number of devices. 1Password doesn't even try to offer a free tier. If you want to use 1Password you pay $36, in advance, for a year to use it.

Bitwarden of the three is the only one to offer an actually useful free plan which allows for the use of any number of devices. And Leo, as you often note when you're talking about Bitwarden, since their free plan actually really does everything you'll probably need, their $10 per year paid plan, at less than a third the price of the others, is mostly just there to support them, although you do get two-factor authentication for that $10 per year.

Leo: Oh, that's worth it. Yeah.

Steve: And I would - yes. And I would argue that for 10 bucks, come on. You know, you should see what my latte costs now. Whoa.

Leo: Incidentally, I did ask when they started advertising, I said, because we'd been burned by LastPass, frankly, I said tell me about the free plan. They said, look, we're open source. It's always going to be free because otherwise people just fork it. It's free free. That's not part of our business model is to make money on the free plans. So that's reassuring. They're not going to pull a LastPass and say, oh, yeah, now you have to pay for it. Open source has a lot of benefits in this area, I think.

Steve: Well, and as Jeremi noted, the icing on the cake is that Bitwarden is also 100% open source.

Leo: Yeah.

Steve: I saw a tweet pass by some time ago that our old friend Alex Neihaus was moving his family to Bitwarden and choosing to self-host a Bitwarden server in his own cloud.

Leo: That's another option, yeah. And actually after this happened I kind of - because for a long time I said, well, I'm never going to be able to protect it as well as professionals at let's say LastPass - whoops - or Bitwarden or 1Password or Dashlane will. But then there's the counterargument which is it's a single point of attack. That's where all the vaults are. So if you self-host, I put it on Dropbox, it's encrypted, they'd have to target me or stumble upon it somehow. And even then they'd only get the encrypted blob that we've talked about that is well-encrypted.

Steve: Yup.

Leo: So I'm thinking about making mine self-hosted now.

Steve: Well, you know, those are the sorts of things that the use of a truly open password manager can provide.

Leo: Can do it, yeah.

Steve: Yes. And while it's not of interest to everyone, I imagine that it's the sort of thing that would appeal to this podcast's audience. On December 7th Bitwarden posted a blog titled "New Deployment Option for Self-Hosting Bitwarden." I've got a link to that in the show notes, bottom of page 10 of the show notes, for anyone who's interested in taking the path that Alex took. And so, you know, that's something that can be done.

And in my digging around over the holidays I stumbled upon a Bitwarden page which linked to their past annual third-party outside network security and application security penetration testing audits. So they're auditing themselves, having themselves audited by a number of different outside firms every year. I ended up not being able to find that same page later, but I found another that provides the same info, and it's under bitwarden.com/help/is-bitwarden-audited. And so there are links there to everything. And I also noted at the bottom of that page that they're signed up with HackerOne to offer and manage bug bounties against their platform.

So the reason I originally chose LastPass and was comfortable endorsing it was that its author opened up its internals to me so that I could understand exactly how it worked. And it was solely on that basis and for that reason that I chose it. I don't regret the decision I made back then. LastPass has been a flawless companion for me over the years. I say that knocking on wood somewhere because my vault has now been absconded with. But as we've observed earlier on this podcast, the world has been changing ever since, and LastPass no longer fits the way it once did. And, you know, it and its organization is beginning to act and feel a bit too old and creaky, which is not what anyone wants in their password manager.

An example of a different more aware and contemporary approach is Bitwarden's description of their management of their user's master password. Bitwarden wrote: "SHA-256 is used to derive the encryption key from your master password. Bitwarden salts and hashes your master password with your email address locally before transmission to our servers. Once a Bitwarden server receives the hashed password, it is salted again with a cryptographically secure random value, hashed again, and stored in our database. The default iteration count used with PBKDF2 is 100,001 iterations on the client - client-side iteration count is configurable from your account settings - and then an additional 100,000 iterations when stored on our servers, for a total of 200,001 iterations by default."

Leo: Yeah. Okay. Can I interrupt with a stupid question? What's with the 100,001, or the 100,100? They don't like round numbers? They only want them round-ish?

Steve: That is a great question. There's nothing...

Leo: Okay, there's no technical...

Steve: No.

Leo: It's just a random...

Steve: It's not like it's a power of two or something where it's going to be some magic. It's bizarre.

Leo: I think it's prime. Oh, well, yeah.

Steve: And that wouldn't matter, either. Anyway...

Leo: No, I know.

Steve: So Bitwarden is not only hashing 100,000 times on our system, and you can turn that to anything you want - and by the way, I would go with like 350,000. That's pretty much...

Leo: Turn it up, yeah. I'm going to turn mine up.

Steve: That's pretty much where you want to be at this point.

Leo: Okay.

Steve: And then they're getting it, resalting it, and then running it another 100,000 times on their high-speed machine because they can; just, you know, because. So anyway, everybody knows what happened then. I went to bitwarden.com/twit.

Leo: Yay. Thank you. By the way, this is not a paid ad in any respect. This is Steve - all right.

Steve: No. Everybody who's listening to this knows that I chose them because they're open source, they have a useful free tier, and they're as good as anybody else.

Leo: Yeah. Are you going to self-host? You're certainly capable of doing that.

Steve: It looks like you actually have to run a Bitwarden server. You can't just have a Bitwarden file.

Leo: Oh. There is a third-party server that is widely agreed to be much faster. I think it's written in Go. So you could use - you don't have to use - it's open source. Somebody else has written a server that's better than Bitwarden's.

Steve: Nice.

Leo: And most of the people who self-host that I know use that software. I'll find it for you. But, you know, I don't self-host.

Steve: I'm not going to bother.

Leo: No.

Steve: No. I mean, again, the reason we did this in the beginning was that we had really strong local client-side encryption, and we were just using the provider to hold onto our data for us.

Leo: Right.

Steve: Unfortunately, LastPass has just demonstrated that we can no longer trust them to do that.

Leo: It's literally the worst thing that could happen to a password, except for not encrypting it right. But it's [crosstalk] good.

Steve: Well, [crosstalk].

Leo: It's as bad as you can get.

Steve: Yes. Well, the reason we were all using the cloud is if your local encryption is solid, it doesn't matter if they lose control.

Leo: Right. Right.

Steve: Unfortunately, they both lost control and their local encryption was not being kept up to date.

Leo: Yeah.

Steve: Okay. So I signed up for Bitwarden, started the $10 plan. Then I went to a menu item I had never used in my LastPass vault, "Export," and I exported a 77 KB CVS file. I opened my shiny new Bitwarden web interface, and under "Tools" at the top was a menu item, "Import Data." From a dropdown menu there, I selected the import source as being "LastPass CSV," provided the filename of the file I'd exported from LastPass, and watched a perfect error-free transfer of my entire legacy LastPass data into Bitwarden. My password database, auto-fill credit cards, and all my secure notes made the move without incident or complaint.

Leo: I have done this, as well, several years ago, but I did it. The only thing I would say is immediately securely erase the CSV file because it's in plaintext; right?

Steve: No, no, no.

Leo: It's not?

Steve: It is an encrypted vault.

Leo: It's still encrypted. Okay.

Steve: Oh, wait, no, no.

Leo: It has to be decrypted, yeah, yeah.

Steve: Yes, yes, in order to import it, of course.

Leo: Treat that carefully.

Steve: Yes. I was briefly confused with the XML that we grabbed from LastPass.

Leo: That's different. Yeah, yeah. No, but this - you have to unencrypt it so it can be reencrypted [crosstalk].

Steve: Yeah. And in fact I remember being very nervous about the fact that that file was on my system. It's like...

Leo: Yeah, delete it securely. The other thing, when I did the import, I had some very long and weird notes in LastPass, and a few of the very long note fields got munged on the import. Or maybe on the export. I'm not sure at what end. But you might check, if you use secure notes, to make sure everything got in. I remember that I had some trouble with that. In fact, actually I think the import didn't work because Bitwarden said, well, I don't know, but this is too long, I can't import that.

Steve: Yeah. And I was wary of that, but I had no complaints or issues at all.

Leo: No problems, okay.

Steve: Simon Zerafa, who is a frequent contributor to the podcast, he said that the Android client put him off a little bit. But you certainly are an Android user.

Leo: Oh, no, no, it's fine.

Steve: Okay.

Leo: Look, I mean, I guess we should also say, and we talked about Tavis Ormandy saying this, that the JavaScript stuff like the plugins for the browsers, perhaps the mobile clients for all these password managers, these are a little bit potentially problematic, especially if code got injected, malicious code got injected and so forth; right?

Steve: Well, again, there is no guarantee being made for any of this over on the client side.

Leo: Right.

Steve: That isn't something we have any control over. If something evil gets in your machine, you know, it can be logging your keystrokes.

Leo: That's a good point.

Steve: It could be watching you type your master password.

Leo: Yeah, all bets are off. But some might say, Tavis did, that it's better to use the binary as opposed to the JavaScript plugin on the browser. And LastPass, Bitwarden, Dashlane, and 1Password all offer a standalone app. I think in some cases it's Electron, so maybe that's not great. The other thing I would say is if you are in the minority, and you only use Apple products, Apple has a very good, I think, password manager that will do all the things a password manager will do, and it's all kept within the Apple ecosystem. It's a little hard - you can use it on Windows and Android, but you have to have an Apple app to do that. Probably should be added to the list of potential candidates for people who are mostly Apple focused.

Steve: I was going to say, yes, if you were a Mac user, then it would make certainly sense to stay there.

Leo: Yeah.

Steve: Of course I'm not. I just [crosstalk].

Leo: I use them. It's nice. It has a very nice generator. It fills it in. But I still, I need Bitwarden because I use Linux, you know, I use other stuff. And I need a cross-platform one. And I, you know, as much as I love Bitwarden, and they are a sponsor, and it's what I use personally, I think you'd be fine with Dashlane or 1Password, as well. I see no reason not to [crosstalk].

Steve: Yes. As I said at the beginning, or when we were talking about this, I see lots of tweets from our listeners who are 1Password users.

Leo: Very happy.

Steve: You know, even Jeremi, when he's not frothing, thinks that 1Password is a good thing, too.

Leo: Right.

Steve: So we have one last thing to talk about, which is the remediation of any danger which may arise from any prior use of LastPass. When I was deciding upon the title for today's podcast I was tempted to title it "Don't Panic," with a nod to Douglas Adams, because I doubt that anyone should panic. There are some caveats, though. As I noted earlier, you should definitely check your LastPass iterations to assure that it had been bumped up to the 100,000 level. And in my show notes here I said please shoot me a note if you discover that it's not 100,100. And that happened immediately. We got a lot of 5,000s people out there. So that's a concern.

But assuming that you have 100,000-plus iterations, and that you're using a master password with good entropy, other than some incidental personal information disclosure of the sort that, you know, as was said, commercial entities that you work with on the Internet probably already have about you, and like all the advertising companies out there that are tracking us, your actual risk of having your own vault decrypted should be low.

If you were using a very high-entropy password with 50, five zero, bits of entropy, a single GPU attempting to crack the LastPass default, assuming that it is 100,000-plus iterations, of an SHA-256-based PBKDF2 would require 200 years and an estimated cost of $1.5 million. Okay. Now, 50 bits is a lot of entropy. Studies have shown that the average password only contains somewhere around 40 bits of entropy. Since that's 10 fewer bits, the strength is 2^10 weaker. 2^10 of course is 1,024, so roughly a thousand. So a 40-bit password would be about 1,000 times less strong. So that brings us down to a couple of months for a GPU to crack a single user's vault at a cost of about $1,500. And this of course reminds us why the strength and the length of our password is so crucial for avoiding brute force attacks. Everybody remember Password Haystacks? It taught us the lesson of how easy it was to create long and highly brute-force-resistant passwords.

The second factor mitigating our risk, if you'll pardon my pun, is the presence of second-factor authentication. Everyone who's listening to this podcast almost certainly has a time-based authenticator and has added the requirement for its use to their most important online accounts. For iOS I still prefer OTP Auth which uses iCloud for its synchronization among my iOS clients. As I scroll through all of the accounts I have registered in OTP Auth, it's comforting to see that I have that additional layer of protection beyond what was contained in my LastPass vault.

But this leaves the theoretical risk that any crucial high-value credentials that might have been saved in that vault which are not protected by a second factor could be at risk. Since LastPass was not encrypting our email addresses and website URLs, there was definite leakage of who we are and where we go and what we do, without any need to decrypt anything. This opens LastPass's users to the potential for potent phishing attacks by leveraging what might be learned from an examination of the unencrypted data contained in their vault. So everyone needs to be on heightened alert, unfortunately forever because this data is never going to go away, for convincing-looking online scams sent to your email addresses and referring to websites you use, which I assume we're all going to be seeing once our LastPass vaults have been deobfuscated by the little utility programs that hopefully our listeners are in the process of writing right now.

The final takeaway is that, if you're concerned that your LastPass master password was not high-quality, high-entropy, and may not have had a high iteration count at the time of its theft from LastPass, the risk of brute forcing might be higher for you. So it might be worthwhile for you to take the time to scan through your vault, after importing it into your next password manager, and manually changing the login passwords of any of your important accounts which are not also protected by some form of strong second-factor authentication.

And finally, the last page of today's show notes contains a link appendix for anyone who's interested in reading the original source material that I found and shared above. I've got Jeremi Gosney's rant link, Wladimir Palant's four recent LastPass blog posts, and also

Steve Thomas's identity over at Twitter and at Mastodon, as well as his website. So anyway, it was a good run. But it was time to say goodbye. And I sincerely hope that none of our listeners were caught by this breach. I know that some of our listeners did have 5,000 iterations. We can hope that they had a really good brute force. And remember, you know, we're talking tens of millions of customers in this backup blob. The danger is much greater for people who are likely to be targeted than our aunts and uncles who we talked into using LastPass 10 years ago.

Leo: Right. And honestly, I would even say, if it was hard to get them to use LastPass, don't feel obligated to move them off of it. Although, yeah, I mean, it would be nice. Incidentally, Steve, are you going to go through all your passwords and change them? Because...

Steve: I'm going to look at them.

Leo: You took them all over with you. And those are still in that blob.

Steve: Correct.

Leo: And presumably if somebody was - I'm sure you had a very long, strong password, and you used two-factor. Does two-factor help in the case of brute force? I guess it does; right?

Steve: Now, that's a really good question. It's the way LastPass implements their so-called, you know, no password deal is that they locally encrypt the decrypted password, the decrypted master password. And then that gets sent to them. When you then use their two-factor authentication for their password-free mode that they came out with a few months ago, they verify that the second factor is correct. Then they send you the encrypted password for you to decrypt locally in order to get access to your vault. So I don't - the way I read that, it doesn't get factored into your master password used to encrypt your vault.

Leo: Wow.

Steve: So the second factor there is not a benefit.

Leo: Wow. So better hope you had a good random, well, it probably isn't random because you had to memorize it, but a good, long, un-brute-forcible master password.

Steve: Yes.

Leo: And given that these guys have had, potentially have had this for some time, and will have it forever more...

Steve: Ever. And we know that cracking technology is only ever going to get better.

Leo: Oh, boy.

Steve: It's not going to get worse.

Leo: This is so bad. So bad. It probably would be prudent, honestly, to go through and change at least the passwords you care the most about, like your bank, credit cards, stores. Those are things they're going to go after.

Steve: And really, I mean, it's not that big a hassle; right? You log onto the site. And that's easy. And then you go to the Change My Password, and you put in the old one, and you make up a new one. I do think, unfortunately, it's probably, I mean, in order to rest easy; right?

Leo: Right. The other thing you could try is set up your new password manager without importing any passwords. And then when you go to a site you know you've got a password, this forces you to say, I forgot, and reset it. And you will gradually fix your passwords by starting fresh with a password manager. That sounds like fun. But it really, honestly, this is why this is so serious. That's kind of the best and only thing you can do if you have to - if you really want to be safe. Hopefully your parents didn't listen to your advice, they didn't get LastPass, they just got the little book at Kmart that said Passwords, and they wrote them all down, one by one, carefully.

Steve: And it's got the rubber band holding it closed.

Leo: Yeah, yeah. And they keep it in the lower left drawer where no one will look. Because they'd probably be safer if they had. Wow. I apologize. When we started doing ads for LastPass, they were a quality product. We were very proud to be associated with them. They saved us in the year 2020. I don't know if TWiT would still be here if they hadn't bought studio naming rights. I apologize to all of you who started using LastPass because of us. But we all got bit. And there's the answer. There's the show you've been waiting for. I've been waiting for.

Steve: Well, at every point we were being clear that the quality of the master password is paramount.

Leo: Yeah.

Steve: And that was the idea; right? You only have one that you need to, like, be in charge of, be responsible for. Make it a good one.

Leo: Right.

Steve: And Password Haystacks teaches us the lesson that you can just put a bunch, like put 20 periods after it, and that makes it way stronger, way more impossible to brute force. And it's, you know, it's not a difficult thing to do.

Leo: Well, in fact, that is the advice somebody gave. If you are writing your passwords down and putting them in a little book, is that's not the password by itself. Add a 10-digit number that you know well. Always, you know, write it down. And that will not only lengthen the password appropriately, it will also make it useless if the bad guys find the book. Wow. Thank you very, very, very much, Steve, for doing this. I think I tooted on Mastodon, dang, the one week we don't have a Security Now!, and we have a story that we really want to know what Steve thinks. But now you know. Thank you, Steve. Do get the show notes for all those links. Read more.

GRC.com is his website. You'll also find two unique versions of the show there. He's got 16Kb audio for people who really have limited bandwidth, and transcripts written by one of the people who has limited bandwidth. That's why the 16Kb show audio exists, so Elaine Farris could download it at her limited capability ranch. She is a - what do they - a collier as well as a court reporter. She does horse shoeing. So she is transcribing this. I believe it's collier. I hope I'm right, Elaine. If I'm not, can you replace it with the right word? Thank you. [Note: Elaine is not a court reporter, not a collier, not a farrier, and has sufficient bandwidth for her full-time occupation as transcriptionist.]

Those transcripts are available at GRC.com. While you're there, pick up SpinRite, the world's best mass storage maintenance and recovery utility, getting better with 6.1, which will be out soon. And if you buy 6.0 now, you'll get 6.1 for free when it comes out. You also can participate in the beta period. I'm sure there will be something like that.

After the show you can go there and leave comments for Steve at GRC.com/feedback. He's still on Twitter. He's going to be - please turn out the lights when you leave, @SGgrc. Actually, don't worry because I think the power company will turn them off, sooner than later. @SGgrc on Twitter. His DMs are open. You can leave comments there, as well. We have the show at our website, TWiT.tv/sn. That's where you can download shows. You can also go to the YouTube channel. There's a dedicated channel. You can find that and all our dedicated channels at YouTube.com/twit. YouTube.com/twit. Subscribing is probably the best way to get it.

Copyright (c) 2014 by Steve Gibson and Leo Laporte. SOME RIGHTS RESERVED

This work is licensed for the good of the Internet Community under the Creative Commons License v2.5. See the following Web page for details: http://creativecommons.org/licenses/by-nc-sa/2.5/