Enpass Security Whitepaper

version 6.5

Enpass Technologies Inc.

September 10, 2020

Contents

Enpass Security Whitepaper

Principles

Here at Enpass, our mission is to keep password management simple across all the platforms with utmost security to data. Every step we take is based on following principles:

- **Don't keep user's data-** Enpass is designed to work locally on your device even without internet connection. To use Enpass, you don't need to sign-up with us. Your data always remains on your device and never leaves it until you explicitly choose so by syncing through any of your cloud accounts, and then too it's encrypted.
- **Always use proven technologies-** Data in Enpass is fully encrypted by 256-bit AES encryption with 100,000 rounds of PBKDF2-HMAC-SHA512 using the peer-reviewed and open-source encryption engine SQLCipher, providing advance protection against brute force and side channel attacks.
- **Only user can access his data-** User's data is only accessible to him through the Master Password which only he knows, and it is not recorded anywhere in Enpass. So even if someone, somehow gets access to datafle he will never be able to unlock it without Master Password.

Vault

Vault in Enpass is a secure place where all the data is kept in encrypted format. It is well structured and consists of following fles:

- **vault.json:** This fle contains plaintext info about your vault that is required for synchronization.
- **vault.enpassdb:** This is a SQLCipher database which is 100% encrypted using your vault key. It contains all your data except attachments more than the size of 1KB. Bigger attachments are saved in a separate SQLCipher database.
- **<uuid>.enpassattach:** These are SQLCipher database fles for each attachment with size more than 1KB, encrypted by a randomly generated key that is stored in vault.enpassdb.

Vault can be accessed only with a Vault Key, which in turn derived from master password that only you know.

Vault Key

A Vault Key is the cryptographic key that is used for performing the AES encryption and decryption of your data. It is derived from your Master Password (and Keyfle, if used) using 100,000 rounds of PBKDF2- HMAC-SHA512. Along with the master password, you can also use a Keyfle that acts as a second factor for generating vault key.

Master Password (Something you know)

The vault key that is used in AES for encrypting your data is derived from your Master Password which always stays in your possession only. Neither your master password nor any of its derivative is recorded within Enpass (except in case when you enable Quick Unlock or use Multiple Vaults). If you forget your master password, there is no way to recover your data.

Key:le (Something you have)

A Keyfle, if used in addition to Master password adds more randomness in generation of vault key. If your master password is somehow compromised, it makes it mandatory for an attacker to have access to an additional fle that only you have somewhere on your system. Desktop version of Enpass can generate 64-bit random keys for you. Like Master Password, if you lost your keyfle, there is no way to recover your data.

SQLCipher and Enpass

SQLCipher is an open source extension to SQLite that provides transparent 256-bit AES encryption of database fles. One can fnd complete design details of SQLCipher here

And here is how SQLCipher is confgured in Enpass:

1. The encryption algorithm is 256-bit AES in CBC mode.

2. Enpass derives key data from your master password (and keyfle if used) using 100,000 rounds of PBKDF2-HMAC-SHA512 and use it as a raw key for SQLCipher. Each database is initialized with a unique random salt in the frst 16 bytes of the fle. This salt is used for key derivation and ensures that even if two databases are created using the same password, they will not have the same encryption key.
3. Each database page is encrypted and decrypted individually. The page size is 1024 bytes.
4. Each page has it's own random initialization vector. The IV is generated by a cryptographically secure random number generator and is stored at the end of the page. IVs are regenerated on write to avoid reuse of the same IV on subsequent writes of the same page data.
5. Every page write operation includes a Message Authentication Code (HMAC_SHA1) of the ciphertext and the initialization vector at the end of the page. The MAC is checked when the page is read back from disk. If the ciphertext or IV has been tampered or corrupted, the HMAC check will cause SQLCipher to report a problem with the database.
6. SQLCipher does not implement its own encryption. Instead, it uses the widely available encryption libraries like OpenSSL libcrypto. In case of Enpass, OpenSSL libcrypto is being used as encryption provider.

Sync

Enpass optionally lets you synchronize your data across devices though a wide range of supported clouds.

Your cloud always contains a copy of same encrypted data as on your device. A copy of your encrypted data is downloaded on your device where is gets decrypted (locally) for real sync operation to merge changes. Afterwards, it gets encrypted again and uploaded back to the cloud.

In a nutshell, your cloud works only as a storage medium and no cryptographic operation (encryption or decryption) is performed there. All such operations are performed locally on your device and your data never leaves your device in unencrypted format.

So somehow, if an attacker gains access to your Enpass data fle, he fnds it protected with your Master Password and thus making is unuseful for him.

Quick Unlock

Mobiles

Enpass facilitate easy unlocking of data through biometrics authorization using fngerprint or face recognition. However, Enpass still requires the master password to decrypt the database where various OS provided hardware protection APIs help us to protect your master password that can only be revealed after biometric authentication.

Biometric and PIN on iOS

iOS Keychain provides a way to store app-specifc sensitive data that can only be accessed by that app. When you enable Biometric or PIN on your device, Enpass stores an obfuscated version of your master password in iOS Keychain that can only be accessed by Enpass. In any case, your master password does not leave the device; neither during the backup of iTunes nor of iCloud Keychain.

Unlocking through Biometrics uses an addition protection of Secure Enclave. Secure Enclave lock the saved master password in such a way that it can be only accessible after a successful biometric authentication. This is so secure that even if someone gets access to your device passcode and legally adds his fngerprints in the phone, iOS immediately invalidates all the cryptographic keys and makes them unusable.

We have restricted the quick unlocking of Enpass only on the devices having a device passcode. Setting up a device passcode ensures that all the data (including the saved master password) in iOS keychain is protected by iOS itself. When you disable device passcode, quick unlock also gets disabled, removing saved master password.

Also, if fve consecutive attempts to quick unlock are unsuccessful, Enpass erases the master password from the iOS keychain and prompts to enter master password to unlock Enpass. On next successful attempt, it is saved again in the iOS keychain. When you disable Quick Unlock from Enpass settings, Enpass erases the master password from the iOS keychain.

Biometric in Android

Android 6.0 and later, provides a new Fingerprint/Biometric API along with enhancement in Android Keystore which is reliable enough to protect our master password. As soon as you enable Biometric from Enpass settings on your device, we create a Biometric authenticated, Enpass specifc random encryption/decryption key in Android Keystore (accessible to Enpass only) and encrypt your master password using this key and store in private area in the device storage. This solution is highly secure because your master password is accessible only after it gets decrypted using the same key stored in the Android Keystore. And that key (in a usable format) can only be retrieved when you authenticate Enpass with your Biometric. This is so secure that even if someone gets access to your Android device passcode and legally adds his fngerprints in the phone, Android immediately invalidates all the encryption/decryption keys and makes them unusable. Even when in use, Enpass will detect this and will automatically disable Biometric in Enpass settings and removes your master password.

PIN in Android

When locking condition met i.e. inactivity time or when app goes to background, Enpass restricts access to all its features and user have to provide a PIN to access data. This does not actually close the Enpass database but adds just an extra screen for authorization. On wrong PIN entry, database will be immediately closed and a master password will be required. Also, if app is killed by OS in background (some Android devices does it frequently if the Battery Optimization is enabled), Master Password will be required at next start.

Desktops

Enpass facilitates easy unlocking of data similar to mobile platforms through biometrics authorization using platform-specifc APIs and Hardware Enclaves. When no Hardware Enclave is detected, it is advised to always use a master password for unlocking. However, in some cases, convenience wins over and a way to quickly access the Enpass data is required. Enpass does support the quick unlock through Fallback Biometric and PIN but a master password is required every time to unlock Enpass when the application is completely closed locking down the Enpass database.

Windows Hello (Microsoft Store version only)

Modern Windows devices have both biometric and TPM hardware. Thanks to Windows Hello as both can be utilized to solve the purpose reliably enough. As soon as you enable Windows Hello on a supported device, we create Windows Hello authenticated Enpass specifc cryptographic keys in TPM and encrypt your master password using those keys and store in App's private area on-device storage. This solution is highly secure because your master password is accessible for use, only after it gets decrypted using the same keys stored in the TPM of your device. And those keys can only be retrieved when you authenticate Enpass with Windows Hello.

Touch ID on macOS (Apple Store version only)

Unlocking through TouchId in macOS uses the protection of Secure Enclave similar to iOS. Secure Enclave locks the saved master password in such a way that it can be only accessible after successful biometric authentication.

Fallback Biometrics

Quick unlocking through biometrics is available in both Windows and Mac through Windows Hello and Touch ID respectively. When locking conditions are met i.e. inactivity time or when the app goes to background, Enpass restricts access to data without locking down the database and users need to authorize using available Biometric method. If the user fails to authorize using biometric, the database will be closed and a master password will be required next time.

PIN

Enpass on all the desktop systems support the unlocking through PIN and it works the same way as fallback biometric for authorization.

Browser Extensions

Browser extensions are additional software that need to be install in the browser. They communicate through the local Web Sockets with Enpass app. Each extension needs to be paired with user intervention diminishing the chance of information effuence. All information is exchange through an encrypted channel with a common key derived using SRP handshake for each session.

The following precautions have been taken to keep the data secure from any unauthorized access.

- **Extension Validity Check-** When a browser extension tries to connect to main Enpass App, we verify the origin of the connection, it must be the unique identifer of our browser extension and browser will not allow installation of any two extensions with the same ID.
- **Browser Validity Check-** When a browser extension tries to connect to the main Enpass app, we also verify its authenticity by checking its code signature i.e. browser must be code signed with relevant company's certifcate (i.e Chrome is signed by Google). If a browser is code signed and not in our whitelist, an additional confrmation is required whether it should be allowed to connect to Enpass or not. Note: Browser Validity Check is not done on Linux as code signing is not available there.
- **Pairing-** Browser extension need to initiate a SRP handshake with a pairing code displayed. User have to manually enter that pairing code into Enpass. If the pairing code matches, both Enpass app and browser extension will be having a common secret at the end of handshake. Further communication will be encrypted with that common secret. Browser extension will also store a provided pairing key for further sessions. It is stored in extension sandbox. User can opt not to store next session pairing key, in that case browser need to pair for every browser session.
- **Same-origin policy-** To prevent the attacks like sweep-in, Enpass auto-flls only on the web pages matching with the saved domain/host names. Furthermore, Enpass never automatically executes any scripts on page-loading for auto-flling, rather, frst it presents a list of items matching with the same domain/hostname and later executes the script to autofll, once the user has selected the item to autofll with.

Sharing

Enpass comes with a feature that allows users to conveniently share an item with others. Well, your data is totally safe when it resides in Enpass but the security of item may be compromised when shared.

There are basically two ways of sharing the data from Enpass.

Using pre-shared key (Secure)

This is a secure way of sharing items with other Enpass users. You frst need to create a pre-shared key for intended recipient from the advanced settings of Enpass. While sharing, the item can be encrypted with the created PSK that only you and the recipient knows. It is recommended to share the PSK with recipient through a medium different than the used for sharing the encrypted item.

Using pre-de:ned key (Insecure)

The data shared this way is not secure and can be imported by anyone having Enpass. This is because a pre-defned key is used to encrypt it that is known to every Enpass. Although, it takes cares of prying eyes very well, but is not secure.

While using insecure sharing, you better follow these practices:

1. Always use a secure channel/medium for sharing. i.e. No one must listen in on or temper with the medium. Practically, it is hard to fnd such a channel. As a default medium we have enabled Mail and iMessage but you can always copy and paste the encoded URL into your own trusted software.
2. Double check that you are sending it to the correct person.
3. Delete the shared text after being sent and ask the recipient to delete the same as well after importing the item in their Enpass database.

Wearables

Limited processing power, memory, input capabilities and lack of APIs restrict Enpass to perform any true cryptographic operations on the supported wearables. Security of your data on wearables relies on default security provided by OS.

Apple Watch and Android Wearables

Enpass for Apple Watch and Android wearable works only with the main Enpass app, installed on paired iPhone or Android device using Apple Watchkit and Android Wearable Data Layer API respectively. Only those items are accessible on Watch which you manually mark as a Watch item on the main Enpass app. These shared items are stored on Watch and instead of your master password, they're protected by watch OS. However, these stored items do not leave the watch in any case; neither during the backup of iTunes and nor of iCloud Keychain. For extra safety, along with the Watch PIN, you can enable the PIN for Enpass on Watch also. Otherwise, if anyone gets access to your watch, will be able to see the items stored on it without any authorization required. So, one should use Apple watch with great attention and care.

User's Responsibility

Using Enpass on Watch is like carrying your physical wallet. For better security of your data, while using wearables, it is advised to

- Use a good device passcode.
- Use a good Enpass Watch PIN Code.
- Avoid storing unnecessary items on Watch. Only the frequently used short items like locker code etc. must be shared for Watch access.

Miscellaneous

Random Numbers

Enpass uses openssl RAND_bytes for generating cryptographically secure random data.

Pronounceable Passwords

Using unique passwords for each website is frst and foremost step toward security. Enpass comes with an handy password generator built in. It can generate both pronounceable and random passwords. Pronounceable passwords are generated with Diceware methodology using a 14400 word dictionary. More details about Diceware are described here.

Password Strength Estimation

Entropy is a measure of password strength. Enpass uses Zxcvbn for calculation of entropy of random passwords. More details about zxcvbn are here.

If a password is pronounceable, Enpass calculate both Zxcvbn and Diceware entropy and least of them will be used to show strength. Strength meter is calibrated for following corresponding entropy to display values.

Entropy	Strength
<35	Very poor
35-50	Weak
50-70	Average
70-100	Good
>100	Excellent

Pwn Password Checker

Enpass lets you check your passwords against the list of breached passwords managed by Troy Hunt's Have I Been Pwned service. It's a trustworthy procedure, ensuring that your passwords are safe in Enpass and never sent to the internet. It works on the k-Anonymity model where the frst fve characters of your SHA1 hashed password (the 40-haracter hash created from your password) are sent to www.haveibeenpwned.com. In response, it sends the list of all the leaked passwords starting with those same fve characters. Enpass then locally compares the passwords' hash to the list, and if it fnds any matching password, you get a warning that the password has been leaked on the internet and must never be used.