SOTA licenta

* <https://0d105255x-y-https-ieeexplore-ieee-org.z.e-nformation.ro/document/9776045>
  + PassImg
  + Foloseste un o imagine configurata, un master password si niste metadate ale site-ului pentru a genera parole
  + Use Angron2/ PBKDF2 hash algos to convert password to binary and then use base85 to convert binary hashes to high-entropy strings
  + Stocarea locala a unor parametrii
  + This can be the main document
* <https://0d10525bd-y-https-ieeexplore-ieee-org.z.e-nformation.ro/document/8843591>
  + PassMan
  + MD5
  + Foloseste 3 parametrii pentru generarea de parole
  + Stocarea locala
  + This can be a referencing document
* <https://0d10525jn-y-https-ieeexplore-ieee-org.z.e-nformation.ro/document/9657969>
  + Analysis of popular password managers
  + This can be used to reference brute force and online attacks
* <https://0d1052685-y-https-ieeexplore-ieee-org.z.e-nformation.ro/document/9814059>
  + What is wrong with password managers?
* <https://owasp.org/www-project-top-ten/>
  + Top 10 OWASP
* <https://1password.com/security/>
  + 1password
  + 1Password is developed by AgileBits Inc. It supports Windows, MacOS, Linux, Android and iOS. Its latest version is 7.6.801. The authentication mechanism is based on PBKDF2-HMAC-SHA512.
  + The password and other sensitive information are stored in a virtual vault that is stored in the company's server by default. The user can backup it locally. Additionally, there is a local vault which is stored in user's device. The backup files of the vaults can be found in the following path in Windows 10: %appdata%\Local\IPassword\backups\. The backup of local vault is an archive file in zip format. There are two files in it: profile.j sand folder.j s. The parameters of the authentication mechanism is extracted from the file profile.j s. Fig. 1 describes the authentication progress of the master password.
  + As shown in Fig. 1, when a user login into 1 password, he/she will input the master password, the software then extract the iterations, salt, ovkey \_data and ov \_mac from the file profile.js. Note that the ovkey\_data and ov\_mac are encoded by base64 in the profile.j s. The default value of iteration number is 100,000. The iteration number of its history versions is only 500. This is a very giant improvement of security.
* <https://www.lastpass.com/security>
  + LastPass
  + LastPass is a freemium password manager. It supports windows, MacOS, Linux, iOS and Android through a web interface, and it also includes plugins for various web browsers. The authentication mechanism is based on PBKDF2-HMAC-SHA256.
  + The LastPass vault data (e.g., URLs, usernames and site passwords, secure notes, form fill items) is stored online and locally. The local storage location is dependent on the operating system and web browser. The storage location in Windows and Android device is show in Fig 2. Take chrome as an example, the local database is a SQLite file. The parameters of the authentication mechanism should be extracted from the LastPassData table: the hash value (the verifier in Fig 3) from the “key” item, the iteration number from the “accts” item.
  + Keepass
  + KeePass Password Safe(abbreviated as Keepass) is a free and open-source password manager. The official version only supports windows. There are several unofficial versions on other platforms including Android, iOS, MacOS.
  + Like the above two software, Keepass store usernames, passwords, and other information including notes and file attachments, in an encrypted file. The main difference is that the authentication mechanism of Keepass can use a combination of a master password, a key file or the current Windows account credentials. And the encryption algorithms are more flexible. Users' can choose three KDF algorithms: AES-KDF, Argon2d, Argon2id and two encryption algorithms: AES-256, ChaCha20. The iteration number is also can modified by users.
  + The KeePass database is stored in a local file. There are three file format: KDB for keepass 1. x, KDBX3 for 2.x, KDBX4 from 2.35. We found that in the latest version, the selection of KDBX depends on the encryption algorithm: KDBX3 for AES KDBX4 for ChaCha20. An analysis of KDBX3 can be found in [1]. But it has some errors. We will analyze KDBX3 and KDBX4 based on AES-KDF in the following.
  + KDBX3
  + The KDBX3/4 file includes two parts: HEADER and BODY. The HEADER stores the major encryption parameters. The HEADER starts from the file signatures and version. The other items includes 1-byte flag, 2-bytes length and variable length content. The structure of KDBX3 HEADER is shown in Table I.
  + In the AES-KDF algorithm without a key file. The TransformRounds, TransformSeed and the MasterSeed combined with the master password generate the AES encryption key. And Keepass encrypts StreamBytes with the key and EncryptionIV. The ciphertext is stored in the first 32 bytes of the BODY part. The authentication mechanism is shown in Fig. 4.
* <https://password-managers.bestreviews.net/faq/which-password-managers-have-been-hacked/>
  + Articol, password managers hacked
* <https://github.com/denandz/KeeFarce>
  + Tool pentru a sparge KeePass