

**Správce hesel s dvoufaktorovou autentizací**

**Autor:** Michal Žernovič (230923), David Zeman(231304), Oliver Bielik (231229), Vojtěch Vaculík (230338)

**Obor:** BPC-IBE  
**Predmet:** BPC-AKR

Obsah

[Úvod 3](#_Toc85991563)

[Ciele projektu 4](#_Toc85991564)

[Teoretická časť 5](#_Toc85991565)

[Dvojfázové overenie 5](#_Toc85991566)

[Hash 5](#_Toc85991567)

[AES 5](#_Toc85991568)

[Stav riešenia 7](#_Toc85991569)

[Autori 8](#_Toc85991570)

[Záver 9](#_Toc85991571)

[Prílohy 10](#_Toc85991572)

# Úvod

V dobe kedy je potrebný účet pre každú službu ktorú používame sa viac a viac rozširujú tzv. Password Managers. Ide o programy, v ktorých si môžeme uložiť heslá a dostať sa k nim pomocou jedného „master“ hesla. Vďaka takýmto programom môžeme zredukovať rôzne risky – môžeme používať komplexné heslá a nemusíme si ich pamätať, stačí nám master heslo. Tieto programy ponúkajú aj možnosť vygenerovať dlhé a bezpečné heslá a automaticky ich uložiť, takže užívateľ nemá dôvod používať to isté heslo na viac službách zo strachu, že by viac hesiel zabudol.

Prirodzene takýto software musí byť aj dostatočne chránený rôznymi kryptografickými funkciami a algoritmami. V tomto projekte vytvoríme jednoduchú aplikáciu na ukladanie hesiel s patričnými bezpečnostnými opatreniami.

# Ciele projektu

Cílem našeho projektu je dosažení plně funkční aplikace, napsané v programovacím jazyce Python. Aplikace bude sloužit k úschově hesel a bude využívat dvoufaktorovou autentizaci pro přístup k uloženým heslům. Jedním z ideálních typů dvoufaktorové autorizace je využití aplikace od společnosti Google, tedy jejich autentizátor, který po přidání aplikace do telefonu nebo počítače vygeneruje náhodný kód, kterým se provede autorizace. Dále, všechny uschované soubory s hesly budou zabezpečeny dostatečně silným šifrováním. Uživatel bude mít na výběr ze tří různých šifrovacích algoritmů, které budou použity pro šifrování jím požadovaných hesel. Všechny změny provedené v databázi s hesly budou logovány (bude uchováván záznam) a v určitých časových intervalech bude prováděna integrita zašifrovaných souborů za pomocí hashovacích funkcí. V případě efektivního vypracování této aplikace také plánujeme implementovat GUI, které ulehčí práci.

Shrnuli, tedy všechny tyto body, tak naším hlavním cílem je dosažení aplikace, která efektivně a bezpečně uchovává všechna hesla, která se dostanou do databáze tohoto programu. Přístup do této aplikace bude chráněn dvoufaktorovou autentizací, kde si uživatel bude moci vybrat z několika možných variant a použití hashovací funkce SHA-256 pro kontrolu integrity a narušení uložených souborů.

# Teoretická časť

## Dvojfázové ověření

Dvojfázové ověření je v dnešní době velmi rozšířený způsob ověřování, jestli je daný uživatel oprávněný přihlásit se do služby. Typicky to bývá kód, který přijde vlastníkovi účtu pomocí SMS zprávy nebo na e-mail. Dvoufázovost tohoto ověření rozumíme v tom, že je nutné poskytnout dva důkazy potvrzující jeho identitu a to:

1. Znalost – něco, co ví pouze uživatel (typicky uživatelské jméno s heslem)
2. Vlastnictví – již zmíněný kód obdržený pomocí SMS zprávou nebo e-mailem, ale také pomocí připojených aplikací, USB klíčem, průkazem s čipem a dalšími způsoby.

Národní úřad standardů a technologií (NIST) ve směrnici z roku 2016 zavrhl SMS kódy jako validní způsob ověřování, avšak rok na to byl tento způsob ve finální směrnici považované za bezpečné.

## Hashovací funkce

Velmi rychlá operace, která vytvoří z původního vstupu výstup fixní délky (otisk). Hashovací funkce je užitečná nejen pro rychlejší vyhledávání určitých dat, ale také při ukládání hesel výrazně zabezpečuje celý proces. Hashovací funkce musí splňovat určité vlastnosti:

* Odolnost vůči získání původní zprávy
* Odolnost vůči jiné předlohy
* Odolnost vůči kolizi, ve smyslu hashování myšleno jako dva odlišné vstupy, které vyprodukují stejný výstup

V historii se používali různé hashovací funkce, například LM Hash používaný v operačním systému Microsoft Windows – jde o hash, který je už dávno prolomený. Jeden z důvodů prolomení bylo omezení hesla pouze na 14 znaků, a navíc každé heslo nad 7 znaků bylo rozdělené do dvou bloků, což výrazně snižovalo složitost prolomení.

V dnešní době je jedna z nejpoužívanějších hashovacích funkcí SHA-256 (Secure Hashing Algorithm) patřící do rodiny SHA-2. Jde o rodinu hashovacích funkcí, která je zatím považována za bezpečnou. Je nástupcem prolomené funkce SHA-1.

## Šifrovací algoritmy

### DES a 3DES

DES (Data Encryption Standard) je symetrický šifrovací algoritmus vyvinutý v 70.letech.

Na otevřený text se nejdříve aplikuje bitová permutace (IP), která usnadňuje hardwarovou implementaci. Výsledek je následně rozpůlen na dvě 32bitové části L0 a R0, které projdou 16 rundami Feistelova schématu – prohození levé a pravé poloviny. V poslední rundě k tomuto prohození nedochází a na závěr se R16 a L16 spojí a je provedena inverzní bitová permutace (IP-1).

Každá runda obsahuje 4 operace:

1. Bitová expanze – duplikuje polovinu bitů (ze 32 bitů se stane 48 bitů)
2. Přičtení 48bitového rundovního klíče pomocí XOR
3. Nahrazení bajtů pomocí tabulky nazývané S-BOX (ze 48 bitů se opět stane 32 bitů)
4. Bitová permutace

V současné době je možné algoritmus DES prolomit útokem hrubou silou za méně než 24 hodin, proto byla nahrazena algoritmem 3DES, která kombinuje 3 za sebou jdoucí algoritmy DES, avšak oproti zcela nově navrženým algoritmům jako například AES je mnohem pomalejší, proto se od použití ustupuje.

### AES

AES (Advanced Encryption Standard) je v současnosti jeden z nejrozšířenějších šifrovacích algoritmů. Jde o symetrickou blokovou šifru, tj. pracuje s bloky stejné délky o velikosti 128 bitů. Na rozdíl od algoritmu DES používá AES substitučně permutační síť (SP síť). Klíče jsou většinou délky 128, 192 nebo 256 bitů.

Proces šifrování u AES se skládá z 10 kol, které obsahují různé matematické operace. Na začátku je provedena operace XOR s částí našeho klíče a plaintextu (nezašifrovaný text). Pořadí následujících operací v jednom kole je:

1. Substitute Bytes – Nahrazení bajtů pomocí tabulky S-BOX
2. Shift Rows – Posouvání bajtů v řádcích
3. Mix Columns – Bajty jsou po sloupcích pozměněné matematickou funkcí
4. Add Round Key – Přidání další části expandovaného klíče

Jestliže máme klíč o velikosti 128 bitů, proběhne 10 kol, avšak v desátém kole je vynechaný proces Mix Columns.

### ChaCha20

# Stav riešenia

# Autori

Na riešení problematiky sa podieľa skupina 4 ľudí, Michal, Oliver, Vojtěch a David. Michal mal za úlohu vypracovať teoretickú časť dokumentu, ktorá je zameraná na úvod k problematike a jej teoretickú časť, ktorú stručne zhrnul na stranách vyššie. Vojtěch mal za úlohu popísať ciele, ktoré chceme dosiahnuť pri riešení problematiky. David v sekcii stav riešenia popisuje aktuálny stav riešenie, ktorý sa bude počas nasledujúcich týždňov meniť, nakoľko sa projekt bude stále posúvať dopredu. Oliver v časti autori opisuje, ako sa každý člen skupiny podielal na tvorbe projektu „Password managera“, či už priamo pri riešení problematiky v jazyku Python alebo pri písaní štúdie. Riešenie problematiky v jazyku Python sme rozdelili nasledvne. Michal má za úlohu navrhnúť GUI a implementovať funkcie, Vojtěch a David spoločne riešia problematiku zabezpečania pomocou HASH funkcie a AES. Oiver ma za úlohu vyriešiť dvojfaktorové overenie a ukladanie údajov do SQL tabuliek.

# Záver

# Prílohy

Zdroje

[1] NIST Softens Guidance on SMS Authentication [online] 2021 [cit. 23-10-2021] https://www.onespan.com/blog/nist-softens-guidance-sms-authentication

[2] Cryptographic hash function [online] 2021 [cit. 23-10-2021]

<https://en.wikipedia.org/wiki/Cryptographic_hash_function>

[3] Advanced Encryption Standard [online] 2021 [cit. 24-10-2021]

https://www.tutorialspoint.com/cryptography/advanced\_encryption\_standard.htm