# Tema – module criptografice

# BCrypt

**Introducere**

Autentificarea este unul din cele mai intalnite procese in aplicatiile pe care le folosim zilnic. Astfel ca dorim o securitate sporita a datelor pe care le stocam in aceste conturi. Prin urmare, la proiectarea unei astfel de aplicatii, decizia in ceea ce priveste stocarea acestor date sensibile este foarte importanta.

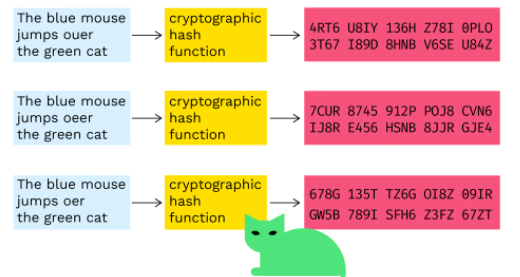
Cea mai intalnita modalitate de a retine aceste credentiale este stocarea lor intr-un tabel: unui utilizator ii este asociata o parola. Atunci cand user-ul isi introduce datele, se va face un request pentru server care va compara parola data cu cele stocate in baza de date pentru a vedea daca exista vreo potrivire.

**Ce este BCrypt?**

BCrypt este un modul criptografic, o functie cu ajutorul careia se poate aplica un algoritm de hash pe o parola pentru a o mentine in siguranta. Acesta a fost proiectat de Niels Provos si David Mazieres in anul 1999. Functia bcrypt sta la baza algoritmului de hash pentru parolele din OpenBSD, dar si pentru toate sistemele Linux. “B” vine de la Blowfish care este un cifru bloc cu cheie simetrica, proiectat de Bruce Schneier in 1993. Acest modul poate fi folosit pentru a stoca parolele in baza de date intr-un mod foarte sigur, fiind implementat in diverse limbaje de programare(PHP, Python, Ruby, Node.js).

**Cum functioneaza BCrypt?**

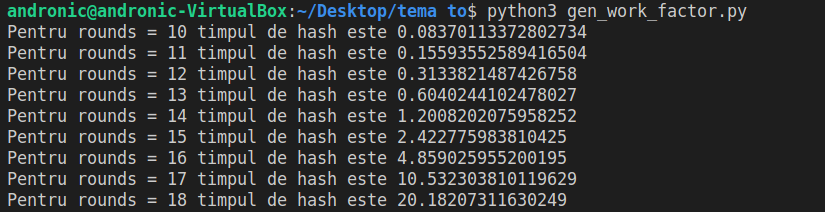
**Functie hash criptografica(CHF):** este un algoritm care pentru date de dimensiunea variabila genereaza un output de lungime fixa. Mai exact, aceasta mapeaza mesajul la o matrice ce are intotdeauna aceeasi dimensiune. Aceste functii nu sunt inversabile. [6]



**“Salt”:** reprezinta un input ce este adaugat de fiecare data cand este stocata o noua parola. Este generat aleator si garanteaza un output unic dupa aplicarea functiei de hash chiar daca 2 parole sunt identice. Parola este stocata in baza de date impreuna cu acest input.[7]

BCrypt functioneaza in 2 pasi: 1. Se genereaza “salt”-ul; 2. Se va aplica functia de hash pe parola cu salt-ul generat.

Desi BCrypt este foarte lent in comparatie cu alte functii destinate securitatii parolelor, output-ul este unul foarte sigur(Thomax Ptacek scrie in articolul Enough with the Rainbow Tables: “The better you can optimize your password hash function, the faster your password hash function gets, the weaker your scheme is”). Rapiditatea calculelor in acest modul depinde de factorul de lucru(work factor) (as putea sa adaug cateva scheme sa vad cum creste si cum scade timpul de procesare.)



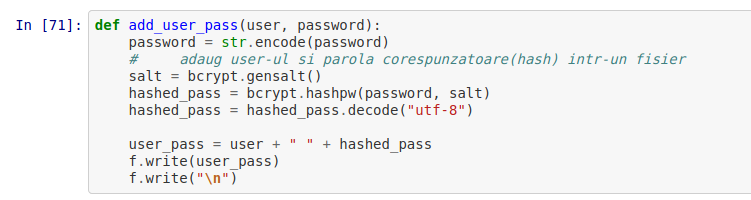
Se poate observa cum timpul creste foarte repede pentru un cost mai mare.

**Cum am folosit eu BCrypt?**

Pentru a folosi aceasta librarie criptografica in python trebuie instalat si importat modulul bcrypt. Am folosit aceasta functie intr-un script care salveaza parola impreuna cu username-ul intr-un fisier si intoarce parola asociata unui user(cripteaza si decripteaza).

In acesta functie dau ca input username-ul si parola.

* Se genereaza un salt.
* Se aplica algoritmul de hash pe parola impreuna cu acel salt care asigura unicitatea criptarii a doua parole ce sunt la fel: acest proces se realizeaza prin functia hashpw(text\_clar, salt).



La decriptare:

* Primesc ca input username-ul si parola
* Verific daca parola pe care am primit-o ca input corespunde cu cea a username-ului dat ca parametru. Aceasta verificare se face cu functia checkpw(text\_clar, parola pe care a fost aplicat algoritmul de hash anterior).



Avantajele:

1. Proprietatea de cascada: o mica schimbare in input va genera o schimbare foarte mare in output
2. Deoarece necesita si generarea unui salt inainte => hashing-ul combinat cu salt-ul protejeaza impotriva atacurilor de tip rainbow.
3. Costul de calcul(the computation cost) este parametrizat, deci poate fi crescut odata cu puterea de calcul de care dispunem. Acest cost mai poarta numele de factor de lucru(work factor). Procesul de hashing este mai lent, acest lucru facand ca un atac prin forta bruta sa fie mai greu de realizat. Insa in acelasi timp, un factor de lucru ridicat, desi aduce un plus de Securitate, poate ingreuna experienta user-ului deoarece se folosesc mai multe resurse pentru a se cripta parola in baza de date(aici trebuie sa reformulez).

Particularitati ale BCrypt:

1. Prefix ajustabil:

Bibliografie(sa adaug si ora si data la care a fost accesat):

1. <https://www.sitepoint.com/why-you-should-use-bcrypt-to-hash-stored-passwords/>
2. <https://auth0.com/blog/hashing-in-action-understanding-bcrypt/>
3. <https://zetcode.com/python/bcrypt/>
4. <https://javascript.plainenglish.io/how-bcryptjs-works-90ef4cb85bf4>
5. <https://pypi.org/project/bcrypt/> (documentatia)
6. <https://en.wikipedia.org/wiki/Cryptographic_hash_function>
7. https://en.wikipedia.org/wiki/Salt\_(cryptography)