Tema 1

Grupa:B5, Groza Vasile

# Exercitiul 1

1. Problema a fost rezolvata in limbajul c++
2. Pentru operatiile de criptare si decriptare am utilizat API-ul specificat in cerinta problemei si anume API EVP
3. In directorul Ex1 sunt urmatoarele fisiere:
4. **file\_service.h**

* Fisierul contine functii de baza pentru lucru cu fisiere (ex citire/scriere) precum si functiile trim si de padare

1. **criptor.cpp**

* Scopul acestuia este de cripta un un fisier de input si scrierea criptarii in fisierul output folosind o cheie random din dictionarul de cuvinte
* Modurile de criptare sunt cele cerute in enuntul problemei si anume ECB si CBC

1. **over\_AES\_service.h**

* Acest fisier h contine functiile de criptare si decriptare in ele doua moduri specificate mai sus
* Sunt implementate atat functile de criptare cat si cele de decriptare
* Ca variabila globala in acest fisier este initial\_vector

1. **main.cpp**

* programul care primeste plaintext-ul, criptotext-ul si modul de criptare si gaseste cheia cu care a fost criptat plaintext-ul.
* In acest fisier se apeleaza functionalitatiile din fisierele descries mai sus.

1. Idea de rezolvare:

* Ideea de rezolvare este una simpla avand urmatoarele date de intrare programul din fisierul main:
  + file\_in (plaintext)
  + file\_out (criptotext)
  + mode (modul de criptare)
  + datele de intrare se valideaza (sa existea file\_in, file\_out si mode sa fie unul din modurile specificate ECB si CBC
* pentru fiecare cheie din dictionar
  + se cripteaza plaintext-ul si se compara cu criptotextul
  + daca acestea corespund inseamna ca am gasit cheia potrivita
* daca cheia nu a fost gasita se afiseaza un mesaj corespunzator

1. Modul de rulare: in directorul Ex3 se afla fisierul MakeFile la rularea caruia
   1. Se ruleaza executabilul criptor pentru a cripta mai intai file\_in cu un mod specificat in file\_out
   2. Se ruleaza executabilul main pentru a gasi cheia cu care sa criptat file\_in si a fost obtinut file\_out de la pasul precedent

# Exercitiul 2

1. Problema a fost rezolvata in limbajul c++
2. Pentru operatiile de criptare si decriptare am utilizat API-ul specificat in cerinta problemei si anume API EVP
3. In directorul Ex2 sunt urmatorele fisiere:

* Directorul **tests** care contine la randul sau 10 directoare in fiecare din care sunt doua fisiere file1.in si file2.in care difera doar printr-un octet
* **main.cpp** care reprezinta rezolvarea problemei propuse si anume
  + functie pentru generarea hash-ului SHA256(**getSHA256**) pentru continutul unui fisier si scrierea acestuia in fisierul specificat ca parametru
  + functie pentru generarea hash-ului MD5(**getMD5**) pentru continutul unui fisie si scrierea acestuia in fisierul specificat ca parametru
  + functie pentru compararea a doua fisere ce contit hash-uri in format hexa
  + si functia carea apeleaza pe acele enumerate mai sus **hash\_and\_compare**
    - aceasta genereza hash-urile SHA256 si MD5 pentru cele 2 fisiere
    - apoi compara hash-urile obtinute si lereturneaza in forma de pointer array de tip int de dimensiunea 2 pe pozitia 0 fiind numarul de bytes egali din hash-ul SHA256 si pe pozitia 1 numarul de bytes egali pentru hash-ul MD5

1. Observatii
   1. In directorul tests se cum am mentionat si mai sus se afla o lista de teste

In main se itereaza prin acete teste si se ruleaza pe fiecare pereche de file1.in file2.in din ierarhiea de directoare si se apeleaza functia **hash\_and\_compare** care returneaza numarul de bytes care difera in hash-urile SHA256 respective MD5 peste acele fisiere raspunsul scriindu-se in fisierul test\_result.txt in urmatorul format:

test0

tests/0/file1.in vs tests/0/file2.in

SHA256 equal bytes : 0

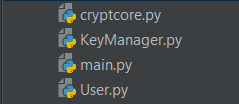
MD5 equal bytes : 0

1. Modul de apelare: se apeleaza fisierul MakeFile care compileaza si executa main.cpp sau se ruleaza executabilul main din acelasi director

# Exercitiul 3

1. Problema a fost rezolvata in limbajul Python ver 3.5
2. Drept librarie criprografica a fost folosita “cryptography”
   1. pip install cryptography //on windows
   2. sau <https://pypi.python.org/pypi/cryptography> pentru surse
   3. <https://cryptography.io/en/latest/> documentatia librariei
3. Exercitiu proriiu zis

Avem urmatoarele fisiere python

1. **cryptocore.py**

* Fisierul contine clasa CryptCore: aceasta ofera functionalitatile urmatoare:
  + Cripta blocuri de plaintext
  + Decripta bloruri de cryptotext
  + modul cheia si iv-ul specificat
* modul de operarea a criptarii si decriptarii plain/crypt textului se afla in acest fisier
  + encrypt\_blocks(args)
  + decrypt\_blocks(args)
  + details there ☺

1. **KeyManager.py**

* Fisierul contine Clasa KeyManager care contine toate cele 3 key
  + Key1 pentru comunicarea ECB
  + Key2 pentru comunicarea CFB
  + Key3 pentru criptarea celor 2 chei
* Acesta are functia get\_key prin care User-ii pot cere cheia necesara pentru comunicare
* Key1 si Key2 vor fi criptate si trimise clientului cu modul ECB pe 128 biti

1. **User.py**

* Fisierul contine clasa User care are urmatoarele atribute:
  + id unic
  + key\_manager pentru a cere cheia de comunicare
  + key3 pentru a putea decripta cheia de comunicare
  + iv initial vector pentru modul de comunicare CFB
  + users o lista cu utilizatorii cu care poate comunica
  + path calea catre sendbox-ul personal
  + crypt\_helper instant a clasei CryptCore la care apeleaza pentru criptare/decriptare a fisierului.

1. **main.py**

* In acest fisier este reprezentat un demo care reprezinta flow-ul general al exercitiului
  + Se genereaza mai intai key1, key2, key3, iv,
  + Se creeaza o instanta KeyManager
  + Se creeaza 3 utilizatori cu datele necesare
  + In directorul exercitiului sunt pregatite deja niste fisiere pentru exemplu de transimtere a acestora intre utilizatori
  + Demo: rulati main.py (detalii in fisierul sursa)