# Security Lab 2

## 2

Programme en c++  cesar.cpp (fais par moi-même, pas très optimisé je précise)

Pour compiler (Unix) :
- Installer gcc
- g++ cesar.cpp -o cesar

Utilisation (MacOS/Unix, après compilation) :

./cesar  (message) , pour le brute force.
./cesar message encrypt/decrypt  key

(Motivation : Je voulais profiter de l'occasion pour m'entraîner au c++)

### 2.1

**Question 1**

Egek guv wp vgzvg ugetgv, pg rcu rctvcigt !

Commande : ./cesar "Ceci est un texte secret, ne pas partager \!"  encrypt 2

**Question 2**

Le chiffrement par Code Cesar n'est pas sur.

Commande : ./cesar "Sl jopmmyltlua why Jvkl Jlzhy u'lza whz zby."  decrypt 7

### 2.2

**Question 3**

Le texte en clair = Code Cesar
Clé = 17
Commande : ./cesar "Tfuv Tvjri"

On regarde juste les 25 lignes de résultat. (simple parce que le message est court)

**2.3**
**Question 4**

**La clé :** 3

**Le message :**

Considerant que la reconnaissance de la dignite inherente a tous les membres de la famille humaine et de leurs droits egaux et inalienables constitue le fondement de la liberte, de la justice et de la paix dans le monde.

Considerant que la meconnaissance et le mepris des droits de l'homme ont conduit a des actes de barbarie qui revoltent la conscience de l'humanite et que l'avenement d'un monde ou les etres humains seront libres de parler et de croire, liberes de la terreur et de la misere, a ete proclame comme la plus haute aspiration de l'homme.

Considerant qu'il est essentiel que les droits de l'homme soient proteges par un regime de droit pour que l'homme ne soit pas contraint, en supreme recours, a la revolte contre la tyrannie et l'oppression.

Considerant qu'il est essentiel d'encourager le developpement de relations amicales entre nations.

Considerant que dans la Charte les peuples des Nations Unies ont proclame a nouveau leur foi dans les droits fondamentaux de l'homme, dans la dignite et la valeur de la personne humaine, dans l'egalite des droits des hommes et des femmes, et qu'ils se sont declares resolus a favoriser le progres social et a instaurer de meilleures conditions de vie dans une liberte plus grande.

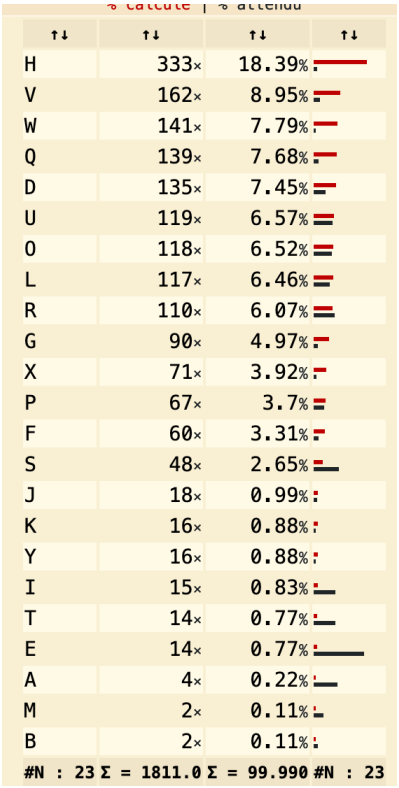
Considerant que les Etats Membres se sont engages a assurer, en cooperation avec l'Organisation des Nations Unies, le respect universel et effectif des droits de l'homme et des libertes fondamentales.

Considerant qu'une conception commune de ces droits et libertes est de la plus haute importance pour remplir pleinement cet engagement.

L'Assemblee generale proclame la presente Declaration universelle des droits de l'homme comme l'ideal commun a atteindre par tous les peuples et toutes les nations afin que tous les individus et tous les organes de la societe, ayant cette Declaration constamment a l'esprit, s'efforcent, par l'enseignement et l'education, de developper le respect de ces droits et libertes et d'en assurer, par des mesures progressives d'ordre national et international, la reconnaissance et l'application universelles et effectives, tant parmi les populations des Etats Membres eux-memes que parmi celles des territoires places sous leur juridiction.

**Méthodologie :**

Grâce au site on effectue une analyse de fréquences. On compare l’analyse obtenue avec le tableau de fréquences fourni en exercice. Puis on remarque que H est probablement équivalent à E. Donc la clé utilisée pour crypter ce message devrait être 3. On teste avec la commande : ./cesar text decrypt 3. On obtien ainsi le messge ci-dessus.



| | | | |
|---|---|---|---|
| H | 333× | 18.39% | |
| V | 162× | 8.95% | |
| W | 141× | 7.79% | |
| Q | 139× | 7.68% | |
| D | 135× | 7.45% | |
| U | 119× | 6.57% | |
| O | 118× | 6.52% | |
| L | 117× | 6.46% | |
| R | 110× | 6.07% | |
| G | 90× | 4.97% | |
| X | 71× | 3.92% | |
| P | 67× | 3.7% | |
| F | 60× | 3.31% | |
| S | 48× | 2.65% | |
| J | 18× | 0.99% | |
| K | 16× | 0.88% | |
| Y | 16× | 0.88% | |
| I | 15× | 0.83% | |
| T | 14× | 0.77% | |
| E | 14× | 0.77% | |
| A | 4× | 0.22% | |
| M | 2× | 0.11% | |
| B | 2× | 0.11% | |
| #N : 23 | Σ = 1811.0 | Σ = 99.990 | #N : 23 |

# 3

## Question 5

```
student@vm04:~/lab2$ openssl enc -aes-256-cbc -d -in secret.enc -out secret.dec
enter AES-256-CBC decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
student@vm04:~/lab2$ cat secret.dec
Vous êtes en train de lire une phrase chiffrée à l'aide de AES 256 en mode CBC
student@vm04:~/lab2$
```

Le fichier contient ce **message** :
Vous êtes en train de lire une phrase chiffrée à l'aide de AES 256 en mode CBC

## Question 6

```
student@vm04:~/lab2$ echo "j'espere faire 100/100 a tous les labs !" >> question6
student@vm04:~/lab2$ open
open     openssl  openvt
student@vm04:~/lab2$ openssl enc -aes-256-cbc -e -in question6 -out question6.enc
enter AES-256-CBC encryption password:
Verifying - enter AES-256-CBC encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
student@vm04:~/lab2$ cat question6.enc
Salted__���M#G�$'\��F�U]��h��.7;Inv��ng�j-�A�J���X1�#�l�?�7student@vm04:~/lab2$ ^C
student@vm04:~/lab2$ openssl enc -aes-256-cbc -d -in question6.enc -out question6.dec
enter AES-256-CBC decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
student@vm04:~/lab2$ cat question6.dec
j'espere faire 100/100 a tous les labs !
student@vm04:~/lab2$
```

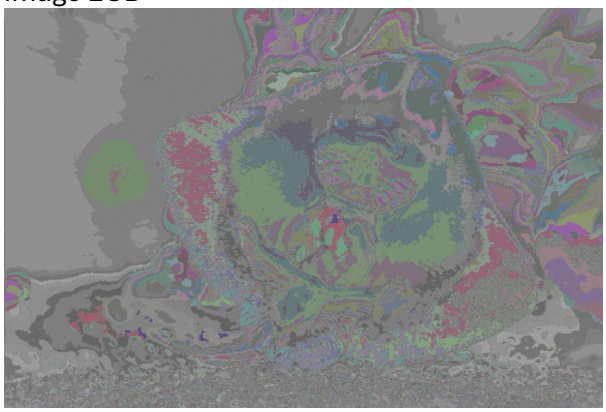
J’ai utilisé comme mot de passe security24.

## 3.1

**Question 7**

On remarque que le chiffrement CBC fait un meilleur travail de chiffrement que ECB. Sur l'image après chiffrement ECB on peut entrevoir la forme de l'animal sur l'image. Alors que sur l'image après chiffrement CBC on a uniquement du « noise ». Cette différence est due au fait que. Le chiffrement ECB traite des blocks de données (dans notre exemple 128 bits) indépendamment les uns des autres avec la même clé, donc si deux blocks sont identiques cela va créer le même block chiffré. Tandis que CBC lui se base sur le block chiffré précédent. Ce qui amène de plus de variabilité.

Image ECB



Lien vers l'image pour le chiffrement ECB :
https://secinf24.pi4.delaage.fr/files/3623d6e9f89ab39bcda994e79f090112a8ec95f959eaaf619048f6e5f1f4d0b3.jpg

Image CBC

Lien vers l'image pour le chiffrement CBC :

https://secinf24.pi4.delaage.fr/files/5d688e4a25150dcedbc58ac10eec560e389b6eb2a676959a25c53a43822e9cac.jpg

## 4

**Question 8**

La première clé contenue dans key.pem est la clé privée et celle contenue dans public.pem est la clé publique. C'est la clé publique contenue dans le fichier public.pem que l'on partage avec les correspondants.

## 4.1

**Question 9**
Commande : openssl rsautl -encrypt -pubin -inkey contact1_pub.pem -in question9 -out question9.enc

**Question 10**
Commande : openssl rsautl -decrypt -inkey mykey.pem -in message.enc -out message.dec

Le text : Bravo ! Vous avez déchiffré ce message

**Question 11**
Commande : openssl rsautl -sign -in question11 -inkey key.pem -out question11.signature

**Question 12**
Commande : openssl rsautl -verify -in message.signature -pubin -inkey contact1_pub.pem -out message.verfie

Le text : Ce message est authentique !

## 5

**Question 13**

```
student@vm04:~/lab2$ openssl dgst -sha256 document.txt
SHA2-256(document.txt)= 3d32af6e200fd2a9fd78c79eb6f301a2ca245e456827ceb8d4610f93b9f59b19
student@vm04:~/lab2$ openssl dgst -sha256 document.txt
SHA2-256(document.txt)= 3d32af6e200fd2a9fd78c79eb6f301a2ca245e456827ceb8d4610f93b9f59b19
```

On constate que le hash reste le même.

**Question 14**

```
student@vm04:~/lab2$ cat document.txt
il faut un système remplissant certaines conditions exceptionnelles, conditions que je résumerai sous les six c
hefs suivants :
1° Le système doit être matériellement, sinon mathématiquement, indéchiffrable ;
2° Il faut qu'il n'exige pas le secret, et qu'il puisse sans inconvénient tomber entre les mains de l'ennemi ;
3° La clef doit pouvoir en être communiquée et retenue sans le secours de notes écrites, et être changée ou mod
ifiée au gré des correspondants ;
4° Il faut qu'il soit applicable à la correspondance télégraphique ;
5° Il faut qu'il soit portatif, et que son maniement ou son fonctionnement n'exige pas le concours de plusieurs
 personnes ;
6° Enfin, il est nécessaire, vu les circonstances qui en commandent l'application, que le système soit d'un usa
ge facile, ne demandant ni tension d'esprit, ni la connaissance d'une longue série de règles à observer.

Auguste Kerckhoffs, La cryptographie militaire, Journal des sciences militaires, vol. IX, pp. 5-38, jan. 1883,
pp. 161-191, févr. 1883
student@vm04:~/lab2$ openssl dgst -sha256 document.txt
SHA2-256(document.txt)= b244651269ef8b00d77c451e55707242143be8e0ce797d0c843eeb745c380285
```

On voit que le hash a changé après modification du fichier.
Le but d'une fonction de hashage et de créer une empreinte unique pour un fichier ou plus généralement une donnée. Cela permet de vérifier par exemple l'intégrité d'une donnée ou son authenticité. C'est aussi un moyen de protéger et cacher des données sensibles.

## 5.1

**Question 15**

```
student@vm04:~/lab2$ openssl dgst -sha256 contrat1.txt contrat2.txt
SHA2-256(contrat1.txt)= cccc5da79fdfb699b8cdf1d79a8d7814fe46e06bde4f201628423495f6e2d195
SHA2-256(contrat2.txt)= 173fb01b24b000789aae6a599193908745b0a031810453a464367c68baa6d333
```

On remarque que les deux hash sont différents

**Question 16**

```
student@vm04:~/lab2$ openssl dgst -md5 contrat1.txt contrat2.txt
MD5(contrat1.txt)= faad49866e9498fc1719f5289e7a0269
MD5(contrat2.txt)= faad49866e9498fc1719f5289e7a0269
```

Ici on remarque que les deux hash sont équivalents bien que les deux fichiers soient différents. Cela pose un problème car désormais nous n'avons pas d'empreinte unique afin de différencier les deux fichiers. Une personne malveillante pourrait par exemple essayer de s'authentifier en utilisant un mot de passe légèrement différent mais produisant le même hash md5. (md5 est obsolète)

```
student@vm04:~/lab2$ diff -q contrat1.txt contrat2.txt
Files contrat1.txt and contrat2.txt differ
student@vm04:~/lab2$
```