Partie 1

Chiffrements symétriques

1.1 Vocabulaire

- **Chiffrer** : transformer les caractères d'un texte pour le rendre incompréhensible, sauf pour celui qui possède la clé de chiffrement.
- **Déchiffrer** : transformer le texte chiffré en texte clair à l'aide de la clé de chiffrement
- **Décrypter** : transformer le texte chiffré en texte clair sans posséder la clé.
- **Cryptologie** : science du secret : possède deux branches
  - **cryptographie** : étude de l'art du chiffrement
  - **cryptanalyse** : analyse des méthodes de chiffrement pour les casser (décrypter)

1.2 à quoi sert le chiffrement ?

Le chiffrement a pour but de protéger nos données, nos communications, mais aussi de signer nos messages et de s'assurer que l'on communique bien avec la bonne personne :

- **Authentification** : L'authentification est la procédure qui consiste, pour un système informatique, à vérifier l'identité d'une entité (personne, programme, machine).
- **L'intégrité** des données désigne le fait que les données ne soient pas modifiées au cours d'une communication ou de leur stockage. Ainsi, si vous envoyez un texte chiffré sur un canal non sécurisé, le texte chiffré pourra être intercepté et altéré par un attaquant avant d'atteindre son destinataire. Pour contrôler cette intégrité, on associe au message une valeur de contrôle.
- **La confidentialité** : Le chiffrement permet de protéger la confidentialité de vos données à l'aide d'une clé secrète.

1.3 Chiffrement par substitution monoalphabetique

Pour ce type de chiffrement, chaque lettre de l'alphabet est transformée en une nouvelle lettre. Et cette nouvelle lettre est unique.

Ici, la fonction utilisée pour le chiffrement de César, est un simple décalage :

$$x \to x + cle$$

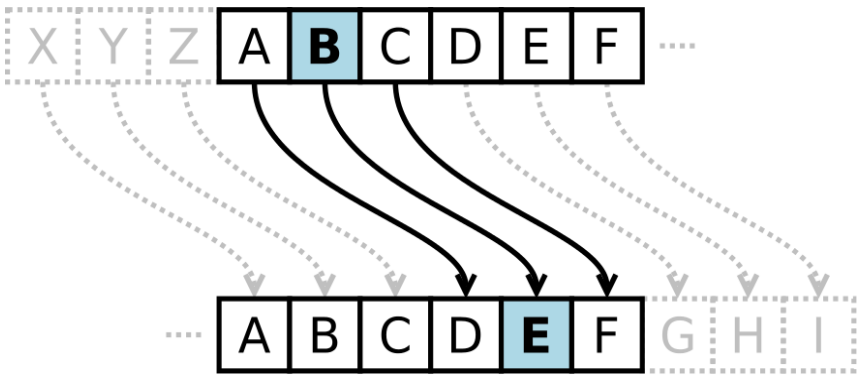

FIGURE 1 – Chiffrement par décalage - wikipedia

Cet algorithme de chiffrement utilise une fonction périodique pour transformer les rangs de chaque lettre :

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

Compléter l'algorithme du chiffrement de César. On suppose qu'il existe une clé `c` correspondant au décalage utilisé. Le message clair est `m` et le message chiffré est `m_chiffre`

```
Pour chaque lettre l du message m:
  l_chiffre = ...
  m_chiffre = ...
```

Combien faudra-t-il faire d'essais, au maximum, pour decrypter ce message par une méthode dite de force brute ? N < ....

Expliquer pourquoi ce type de chiffrement peut-être facilement decrypté par analyse frequentielle ?

1.3.1 Substitution polyalphabétique

Polyalphabetique : Se dit d'un chiffre où un groupe de n lettres est codé par un groupe de n symboles.

- L'exemple suivant montre une polysubstitution simple avec une clé de longueur 3 qui va décaler les lettres de l'alphabet :

On définit la clé '123' qui indique que le premier caractère sera décalé d'une position, le second de 2 et le troisième de 3 positions, etc.

Le mot : WIKIPEDIA donne donc dans ce cas XKNJRHEKD.

- La machine Enigma utilisait ce principe de codage, à l'aide de 3 rotors.
- Le chiffrement *Playfair*, (voir le cours en ligne) s'apparente à une substitution polyalphabetique, puisqu'il substitue des digrammes (groupes de 2 lettres) dans le texte d'origine.

1.3.2 Points communs des chiffrements symétriques

Ces algorithmes fonctionnent selon :

- Entrée
- repetition n fois de substitutions et permutations. Utilise une clé unique pour chiffrer et dechiffrer.
- Sortie

Pour le chiffrement symétrique, le seul secret, c'est la clé de chiffrement.

Partie 2

Exercices

2.1 Substitution monoalphabétique : Code de César

2.1.1 Quelle clé a été utilisée pour chiffer ce texte (avec l'algorithme de César)?

jyfwavnyhwopl hwwspxbll

2.1.2 Décrypter ce message.

2.1.3 Proposer un programme en python qui chiffre un message clair `m` en un message codé selon la clé numérique `c`.

Aide :

- La fonction `ord('x')` retourne un entier correspondant à la position d'un caractère dans la table ascii. Par exemple, `ord('A')` retourne 65, `ord('B')` retourne 66,...
- La fonction `chr(N)` retourne le caractère de rang N : `chr(65)` retourne 'A'.

2.2 Substitution polyalphabétique : Chiffrement de Playfair

A partir des explications données dans la video :

2.2.1 Construisez la matrice pour l'algorithme de Playfair avec la clé *estienne*.

2.2.2 Chiffrer le message : *COUPERLETRANSMETTEUR*.

2.2.3 S'agit-il d'une méthode utilisant la substitution monoalphabetique, ou polyalphabetique ?

2.2.4 Est-ce qu'avec cette méthode, le decryptage peut être facilité par l'analyse fréquentielle ?

2.3 Frequences des lettres dans un texte

La fonction suivante retourne une liste de 26 valeurs de type *float*, donnant dans l'ordre de l'alphabet, la frequence pour chaque lettre dans un texte :

```
1 def freq(m):
2     frequences = [0]*26
3     n = len(m)
4     for c in m:
5     ...
```

- **Question 1** : Compléter le script de la fonction `freq`.

(Suite de l'exercice sur le chiffrement de Playfair)

Soient les deux textes chiffrés suivants :

a. DGFHTMOMEIAMTLMFGOKFGOKEGDDTRWEIAKZGFGFSTEKGOMLASTTIFG FOSTLM FTFGOK MGWMFG OKRTSA JWTWTA WDTFMG FDAOLT WMOSSA FGOKET WKRWFD TEIAFM ROAZSG MOFKOT FFTXAW MLARGW ETWKJW AFROSD OAWSTA WDAMOF HGWKDT STEITK SADAOF

b. YOHGMV MFCBRB GWFNIZ ZPSURW FUOIPU WYITFA NIETGG DOCKAC PQE- BEW PMXEMK VGRAIL KWWXZO CILGPM QOVCGE GMYEBQ RYACJM WX- ULFR VQMDCY LZFYZM YTPCRF DCRJNS FIHIQG RZE-PRC VWMDIL KGYDQO RVFMXM CRCNIM UGRBKR BOOIUG PQCBVZ NEYACE

- **Question 2** : L'un des 2 a été chiffré avec un algorithme de substitution mon-alphabetique, et l'autre, poly-alphabétique. Lequel est mono-alphabétique ?

2.4 Chiffrement symetrique par la fonction XOR

Cet exercice est inspiré du TP cryptographie du site de hmalherbe.fe - NSI

La fonction XOR est la fonction du OU EXCLUSIF.

- **Question 1** : Donner la table de vérité de la fonction XOR
- **Question 2** : Programmer la fonction `xor` en python qui retourne le resultat de XOR appliqué aux 2 paramètres a et b.

Exemple :

```
>>> xor(0,1)
>>> 1
```

On utilisera pour cet exercice les fonctions suivantes, qui permettent de transformer un texte en binaire, à partir du code ascii des lettres :

```
def text2bin(texte):
    """ convertit une chaine de caracteres en binaire
    param:
    texte: str, une chaine de caracteres
    return:
    bin: str, une sequence de 8 bits par caracteres
    chaque sequence de 8 bits est la valeur du code ascii du caractere
    convertie en binaire (sur un octet)
    exemple:
    >>> text2bin('HELLO')
    >>> '0100100001000101010011000100110001001111'
    """
    bin = ''
    for l in texte:
        bits = f'{ord(l):b}'
        # f est une fonction de formatage et convertit
        # une valeur decimale ord(l) en binaire
        # ord(l) donne la valeur numerique correspondant au caractere ascii l
        oct = octet(bits)
        bin += oct
    return bin

def octet(bits):
    """ rajoute les 0 au nombre binaire pour mettre en format de 8 bits
    """
    byte = str(bits)
    for i in range(8-len(bits)):
```

```
28          byte = '0' + byte
29      return byte
```

- **Question 3** : programmer toutes ces fonctions dans un fichier python (ou un notebook). Tester en particulier la fonction `text2bin` sur un texte relativement court (2 caractères). Recopier le résultat obtenu.

La fonction suivante transforme une séquence de chiffres binaires en une valeur décimale :

```
1  def bin2dec(oct):
2      # oct est un str de 8 caracteres
3      n = 0
4      for i in range(len(oct)):
5          n += int(oct[-i-1])*2**i
6      return n
```

Exemple :

```
1  >>> bin2dec('001001')
2  >>> 9
```

- **Question 4** : Utiliser cette fonction pour écrire le script de `bin2text`, la fonction inverse de `text2bin`, qui, pour une sequence binaire donnée, de longueur multiple de 8 bits, retourne la chaine de caracteres correspondants.

Le XOR (OU exclusif) est très utilisé dans les protocoles de chiffrement symétrique. En effet, c'est une opération qui est sa propre réciproque, ce qui n'est pas le cas du ET ni du OU.

On souhaite chiffrer un message (par exemple une chaîne de caractères) à l'aide d'une clé binaire. Seuls des caractères ASCII sont utilisés.

- **Question 5** : Ecrire le script de cette fonction, que l'on appelera `chiffre_xor`. Cette fonction prend 2 paramètres : la séquence de bits à chiffrer, ainsi que la clé de chiffrement. Cette clé est de dimension inférieure ou égale au premier paramètre. Il faudra l'appliquer de manière périodique aux bits de la séquence à chiffrer (voir cours sur le chiffrement poly-alphabetique).
- **Question 6** : Utiliser les fonctions du programme pour :
  - convertir un message secret en une sequence binaire
  - chiffrer cette sequence binaire avec l'algorithme XOR, en utilisant la clé `10101`
  - afficher le message chiffré
  - dechiffrer le message
  - afficher le message dechiffré