Université de Sherbrooke

LABORATOIRE 1

Attaques cryptographiques
INF 808 (A2023)

Auteur :
Fridrich Destin Mortiniera
Matricule #16 042 989

Table des matières

1 CODE CAESAR

1.1 Réponse à la question 1 :

Au vu du code du Listing1, nous comprenons que chaque caractère dans un jeu de caractères (ASCII ou Unicode) est associé à un code numérique unique. Lorsque est implémenté une translation de caractère, comme le chiffrement de Caesar, on effectue un décalage sur les codes numériques des caractères. Par exemple, si on voulait décaler de 3 positions vers la droite, alors le caractère A (code 65) devient D (code 68), B (code 66) devient E (code 69), etc.

Pseudo code d'une fonction qui prend comme paramètres d'entrée un caractère 'a', ..., 'z' et une clé « k » et retourne le caractère translaté :

```
Fonction translater_caractere (caractere, k):
    alphabet <- "abcdefghijklmnopqrstuvwxyz"
    index_original <- position de caractere dans l'alphabet
    index_translated <- (index_original + k) modulo taille de l'alphabet
    caractere_translate <- caractere à l'index index_translated dans
l'alphabet
    Retourner caractere_translate
Fin Fonction

// Vérification de la fonction pour les caractères 'a' et 'z'
caractere_a <- 'a'
caractere_z <- 'z'
k <- 3

caractere_a_translate <- translater_caractere (caractere_a, k)
caractere_z_translate <- translater_caractere (caractere_z, k)

Afficher "Caractère 'a' translaté:", caractere_a_translate
Afficher "Caractère 'z' translaté:", caractere_z_translate
```

En exécutant ce pseudo-code avec une clé de 3, le caractère 'a' sera traduit en 'd' et le caractère 'z' sera traduit en 'c'.

1.2 Réponse à la question 3 :

Le chiffrement et le déchiffrement sont deux concepts étroitement liés en cryptographie. Le chiffrement consiste à transformer un texte clair (message original) en un texte chiffré (message transformé et illisible) à l'aide d'un algorithme et d'une clé de chiffrement. Tandis que le déchiffrement est le processus inverse du chiffrement. Il consiste à prendre un texte chiffré et, en utilisant la clé de déchiffrement correspondante, à le ramener à sa forme originale, c'est-à-dire au texte clair.
La relation qui existe entre le chiffrement et le déchiffrement est que ces deux opérations sont inverses l'une de l'autre. La sécurité du système de chiffrement repose généralement sur la difficulté de déchiffrer le texte chiffré sans la clé de déchiffrement adéquate.

1.3 Réponse à la question 4 :

A- Explication sur 2 clés différents (k et k') donnant un même chiffré

Lorsqu'un même chiffré est obtenu en utilisant deux clés différentes, cela signifie que les clés produisent un effet équivalent lors du processus de chiffrement. Cela peut se produire dans certains systèmes de chiffrement lorsque les clés sont liées d'une manière spécifique. Prenons le cas du chiffrement de Caesar, où chaque lettre est décalée d'un certain nombre de positions dans l'alphabet. Si l'on choisit une clé k pour le chiffrement et une autre clé k' qui est un multiple de la taille de l'alphabet (26 en l'occurrence), on obtiendrait le même chiffré. Par exemple, si k = 3 et k' = 29, les deux clés produiront le même chiffré pour un texte donné.

Déduction du nombre (minimal) de clés qui, pour un texte claire donné, fournissent des chiffres différents

Pour un texte clair donné, le nombre minimal de clés qui fournissent des chiffrés différents dans le cadre du chiffrement de Caesar est égal à la taille de l'alphabet, car si vous choisissez k dans la plage de 1 à 25 (inclus), vous obtiendrez des décalages uniques pour chaque clé, produisant ainsi des chiffrés différents.

B- Explication du pourquoi le code Caesar se prête à une attaque de force brute

Le chiffrement de Caesar est vulnérable à une attaque de force brute en raison de sa simplicité et du petit nombre de clés possibles. Le chiffrement de Caesar implique un décalage fixe dans l'alphabet, il existe un nombre limité de clés possibles, soit autant que la taille de l'alphabet (26 clés possibles). Cela le rend vulnérable à une attaque où l'attaquant essaie toutes les clés possibles jusqu'à ce qu'il trouve la bonne clé.

Comment une attaque de force brute pourrait être menée pour décoder un texte chiffré sans connaître la clé :

1. **En générant les possibilités :** L'attaquant génère une liste de toutes les clés possibles (décalages de 1 à 25 dans le cas de César) et prépare une liste vide pour stocker les résultats.
2. **En testant chaque clé :** L'attaquant chiffre le texte chiffré avec chaque clé possible en utilisant le chiffrement de Caesar. Il compare ensuite le résultat avec le texte chiffré donné.
3. **En comparant les résultats :** Lorsque l'attaquant trouve un texte chiffré correspondant à l'original, il a probablement trouvé la clé correcte. Cela se produit lorsque le texte déchiffré a un sens cohérent.
4. **En décryptant le texte :** L'attaquant peut alors utiliser la clé trouvée pour déchiffrer tout le texte et obtenir le texte d'origine.

Cette méthode de force brute est relativement simple à mettre en œuvre, mais elle nécessite d'essayer toutes les clés possibles, ce qui peut être fastidieux et chronophage. Cependant, en raison du petit nombre de clés dans le cas du chiffrement de Caesar, cette attaque est réalisable même sans ordinateur. De nos jours, avec l'apparition des postes Quantum, il est encore plus facile à un attaquant de venir à bout d'un chiffrement de Caesar.

1.4 Réponse à la question 5 :

- La valeur `clear_text` est « `universite de sherbrooke` »
- La clé K qui y correspond est la « `clé 11` »

1.5 Réponse à la question 6 :

Aux vues du code exécuté dans le Lsiting2 et des tables 1 &2, les caractères dans le codage ASCII standard sont représentés sur 8 bits, ce qui signifie qu'ils peuvent prendre des valeurs de 0 à 255. Cependant, il existe des variations du codage ASCII qui étendent cette gamme pour inclure des caractères spéciaux à d'autres langues ou fins. L'une de ces variations est l'ASCII étendu. Le caractère 'é' a un code de 233 dans l'ASCII étendu (parfois appelé ISO-8859-1 ou Latin-1). Toutefois, cela dépasse la plage des valeurs 0 à 255 dans le codage ASCII standard. Pour représenter ce caractère, deux octets sont nécessaires.

La valeur du code ASCII étendu associée au caractère 'é' est donc 233. Cependant, si l'on travaille avec des encodages plus modernes comme UTF-8 (qui est compatible avec l'ASCII étendu), le caractère 'é' est représenté sur un seul octet, car UTF-8 utilise une méthode de codage variable qui permet de représenter différents caractères avec des longueurs d'octets variables. Cela permet de représenter efficacement un large éventail de caractères tout en économisant de l'espace lorsque des caractères plus courants sont utilisés.

2 CHIFFREMENT PAR SUBSTITUITION

2.1 Réponse à la question 7

En considérant un alphabet où seules les lettres minuscules a-z sont utilisées et que chaque lettre est codée sur 16 bits, alors nous avons 2^{16} (= 65536) valeurs possibles pour chaque lettre de l'alphabet. Cependant, il y a une restriction : la valeur prise par une lettre doit être différente de celle prise par toutes les lettres précédentes.

Nombre de possibilités pouvant être évalué lettre par lettre :

- La lettre 'a' peut prendre n'importe laquelle des 65536 valeurs possibles.
- La lettre 'b' peut prendre n'importe laquelle des 65536 valeurs possibles, sauf la valeur déjà prise par 'a'. Cela signifie qu'il y a 65536 - 1 = 65535 possibilités pour 'b'.
- La lettre 'c' peut prendre n'importe laquelle des 65536 valeurs possibles, sauf celles déjà prises par 'a' et 'b'. Cela signifie qu'il y a 65536 - 2 = 65534 possibilités pour 'c'.
- Ce processus continue pour les lettres suivantes jusqu'à 'z'.

Pour calculer le nombre total de clés possibles pour cet alphabet, nous multiplions simplement le nombre de possibilités pour chaque lettre :

> = Nombre total de clés possibles = Possibilités pour 'a' * Possibilités pour 'b' * ... * Possibilités pour 'z'
> = 65536 * 65535 * 65534 * ... * 3 * 2 * 1

C'est un calcul factoriel, et le résultat sera un nombre très grand, bien au-delà de la portée d'une attaque de force brute réaliste. Une telle attaque serait extrêmement longue et nécessiterait des ressources considérables pour essayer toutes les clés possibles, la rendant pratiquement inenvisageable.
En résumé, avec un codage sur 16 bits par lettre de l'alphabet, le nombre de clés possibles est tellement élevé qu'une attaque de force brute n'est pas envisageable en pratique.

2.2 Réponse à la question 10

Pour conjecturer le texte en clair qui correspond à « `cipher_text_1` », nous pouvons comparer la liste ordonnée des caractères obtenue à partir de « `cipher_text_1` » avec la liste ordonnée des caractères obtenue à partir de « `clear_text_hugo` ». La supposition repose sur l'idée que les caractères les plus fréquents dans le texte chiffré correspondent probablement aux caractères les plus fréquents dans le texte en clair. Voici les étapes correspondantes :

- Tout d'abord, on utilisera la fonction « `freq_text` » pour obtenir les listes ordonnées des caractères et de leurs fréquences à partir des deux textes, « `cipher_text_1` » et « `clear_text_hugo` ».

- Puis, il faudra comparer ces deux listes pour voir si on pourrait faire correspondre les caractères en clair aux caractères chiffrés en fonction de leurs fréquences

Cette approche suppose que les caractères les plus fréquents dans « `cipher_text_1` » correspondent aux caractères les plus fréquents dans « `clear_text_hugo` ». Elle utilise les fréquences des caractères pour essayer de faire correspondre les caractères en clair aux caractères chiffrés. Cependant, il faudra garder à l'esprit que cette méthode de correspondance n'est pas infaillible et peut ou ne pas donner de résultats précis si le texte chiffré a été chiffré avec une méthode plus complexe que le chiffrement de Caesar.

2.3 Réponse à la question 12

À venir

2.4 Réponse à la question 13

À venir