Compte Rendu
TP1 – Réseau et Sécurité

Suzeau François

Liens github du code source :

https://github.com/FrancoisSuzeau/Vegenere.git

I. Introduction

L'objectif de ce TP était d'implémenter un chiffrement de Vigenère, puis de l'analyser selon différente méthodes.

Le programme est écrit en C++ dont vous trouverez les fichiers source répartit dans différents modules répondant plus ou moins aux étapes du sujet. Un Makefile est disponible à la racine du projet si vous souhaitez recompiler le programme après l'avoir modifié (pour par exemple utiliser un texte fournit dans le README).

Ce programme a été développé sous Windows mais je ne vois aucun problème à l'utiliser sur un système Linux puisque qu'il n'utilise que les librairies standards.

II. Architecture et réponses aux questions

Il y a deux modules : Le module VigenereCrypto s'occupe de la capture du texte, de la clé, de la transformation en une chaine de caractère acceptable par le programme (la classe Input) puis il encrypte le texte avec la clé selon le chiffrement de Vigenère (la classe Encryption).

Le module Analysis s'occupe de l'analyse du texte chiffré par les différentes méthodes demandées dans le sujet.

a) La classe Input

Cette classe a pour objectif de répondre aux questions 1 à 3. Il est demandé dans le sujet d'afficher chaque résultat aux questions mais pour des soucis de lisibilité sur le terminal, ces réponses ne sont affichées qu'après la transformation en texte acceptable.

On commence le programme par un texte directement écrit en français avec accents, espace et caractères spéciaux. Chacun de ces caractères reçoit un traitement particulier :

- Les caractères compris entre 33 et 96 de la table ASCII sont supprimé.
- Le caractère « ~ » est supprimé.
- Les caractères « é, è, ç, à, ù » sont transformé en « e, e, c, a, u ».
- Les caractères « §, µ, £, ¨, ¤, € » sont remplacé par un espace
- Les espaces sont supprimés (donc les caractères précédent).
- Les majuscules sont mises en minuscule.

Il est possible de directement écrire son propre texte mais en enlevant le commentaire et en recompilant le programme.

Il est demandé ensuite d'entrer une clé qui recevra le même traitement que le texte.

b) La classe Encryption

Cette classe répond aux problèmes posés dans les questions 4 à 5 à savoir récupérer le texte et la clé transformé puis de chiffrer le texte avec la clé dans la méthode VigenereEncryption avec la formule suivante : (dans la méthode transformLetter)

n = (m + b) modulo 26

avec n la lettre chiffré, m la lettre du texte original à la position i et b la lettre de la clé à la position j donné dans la boucle.

La méthode VigenereDecryption inverse le processus en déchiffrant le texte cypher selon la clé avec la formule suivante : (dans la méthode reverseLetter)

n = (26 – b + m) modulo 26

c) La classe Analysis

Cette classe s'occupe de répondre aux questions 6 à 14 du sujet. Elle commence par calculer les occurrences de séquences de taille supérieur ou égale à 3 dans le texte chiffré. Pour les même raisons ces occurrences ne sont pas affichées dans le terminal (mais le code est disponible à la ligne 78 de la classe Analysis).

On utilise ces calculs pour ensuite calculer la clé selon la méthode des diviseur commun (la méthode calculateKeyLength). A noter que la méthode va choisir le maximum d'occurrence de ces diviseur commun donc si la clé est de taille paire elle choisira forcément 2 comme taille de clé. Pour cela, il est laissé à l'utilisateur quelle taille de clé il souhaite garder en indiquant tous les choix possibles et les occurrences calculé.

Réponses de la question 9 :

Puisque nous avons un texte Tr composé seulement de lettres minuscules il y a donc 26 lettres possible. Obtenir une lettre donnée est de 1/26 chance donc

obtenir une paire identique est égale à 26 x $(1/26)^2$ = 1/26 qui est environ égale à 0.0385.

Réponse de la question 10 :

On obtient des résultats différents pour l'anglais car la distribution des lettres dans un texte anglais n'est pas les mêmes que pour le français par exemple. A noter aussi que cette probabilité peut varier selon la qualité de la source (langage soutenu ou commun) ou même de l'auteur (ex : La disparition de Geoges Perec).

On appelle ensuite la méthode FriedmanTest qui calcule une nouvelle fois la longueur probable de la clé.

On demande ensuite quelle longueur de clé l'utilisateur souhaite utiliser pour l'analyse fréquentielle.

Celle-ci se fait en décomposant le texte chiffré en sous séquence de la taille de la clé, puis en récupérant chaque lettre de chaque sous séquence à une position (on itère de 0 à la taille de la clé – 1) on affiche la fréquence d'apparition de cette lettre ainsi que la fréquence d'apparition de toutes les lettres dans un texte français ou anglais.

L'utilisateur choisi quelle lettre il veut échanger, et la lettre correspondante dans un texte d'une certaine langue, le programme fait son calcul et affiche la lettre de la clé comme étant possible dans la position courante.

Une fois la taille de la clé entièrement parcourue on obtient la clé que l'on teste sur le programme chiffré et on nous dit si nos prédictions était bonne ou non en nous laissant un petit cadeau.

d) <u>La classe Bazeries</u>

Cette classe s'occupe de répondre aux question 15 et 16. Elle commence par récupérer le texte chiffré puis de demander à l'utilisateur de choisir son mot probable.

Puis on commence l'itération à 0 et on récupère la sous chaine de la taille du mot probable dans le texte chiffré (méthode likelyWordAtPos), on parcourt cette sous chaine une à une et on soustrait la lettre correspondante à la lettre du mot

probable à la même position. Cela nous donne une distance que l'on utilise comme indice pour trouver lettre résultante dans l'alphabet. Une fois toute la séquence parcourue, on recommence en incrémentant de 1 la position de départ de la séquence à récupérer dans le texte chiffré.

On laisse le choix à l'utilisateur d'arrêter ce procéder si il pense avoir vu la clé dans la séquence retourné.

III. Bilan

La plus grande difficulté rencontrée fut la partie sur le test de Friedman. Je n'ai pas réussi à comprendre qu'elle était son fonctionnement ni son intérêt ce qui m'a empêché de répondre à la question 13. Cette méthode n'est en effet pas digne de confiance car elle calcul toujours une taille inférieure à la clé et semble énormément varier en fonction de celle-ci.