Introduction :

Bien que la cryptologie moderne s'est scindée en deux catégories : la cryptographie et la cryptanalyse. La cryptographie est une discipline visant à garantir la confidentialité, l'intégrité et l'authenticité lors d'échanges d'informations entre deux parties. Elle a des applications dans plusieurs secteurs civils et militaires. D'ailleurs, la sécurité du commerce électronique ainsi que la protection de la vie privée reposent de plus en plus sur cette science.

La cryptographie à clé secrète (ou symétrique) est la forme la plus ancienne de chiffrement. Elle suppose qu'une clef est a priori partagée par les deux parties, sa souplesse et sa facilité d'adaptation aux besoins concrets de l'industrie ont permis de rester très active.
Les chiffrements à clef secrète ne sont pas pour autant infaillibles. En effet, un crypto-système sans faille, du point de vue mathématique, est concevable. Cependant, sa grande complexité du point de vue implémentation et son coût d'utilisation le rendent prohibitif. Il ne serait donc pas d'une grande utilité pour l'industrie.

Chercher les failles de ces systèmes "imparfaits" est donc le but de la cryptanalyse. Cette disciplineregroupe tous les moyens de casser les crypto systèmes sans avoir aucune idée de leurs fonctionnement interne (boite noire), et de retrouver des messages sans posséder leur clef en utilisant tous ces failles on va étudier une boite noire (crypto-système symétrique) simple, faible, et fortement inspiré d'un système réel et de mettre en évidence ses faiblesses. on disposera d'un certain nombre de triplets (clair,clé,chiffré).

Organisation du document

Nous commencerons ce rapport par une introduction aux crypto-systèmes symétrique.
On décrira ensuite les règles qui rendent la relation entre message claire et son chiffré et la clé complexe, puis on concernera la cryptanalyse.
La deuxième partie sera conçue sur la partie pratique ou on commence par le plan de travail, puis le langage qui sera proposé pour l'implémentation, ensuite l'interface graphique et les fonctionnalités implémentées dans la boite à outil.
Pour finir, nous dévoilerons les résultats de notre attaque sur la boite noire.

I. Conception

1. Présentation générale des chiffrements

La cryptographie utilise un chiffre pour coder un message. Le déchiffrement est l'opération inverse, par une personne autorisée à retrouver le message clair.

Un cryptosystème est la donnée:

- d'un ensemble fini M de messages (textes) clairs

- d'un ensemble fini C de cryptogrammes (messages cryptés)

- d'un ensemble fini K de clefs

- d'une application d'encryptage

e : M K ! C

et d'une application de décryptage

d : C K ! M

telles que, pour tout M 2 M et tout K 2 K

d(e(M; K); K) = M

-Crypto-système symétrique

Si e = d, la clef K = e = d est dite symétrique de même que le crypto-système

et on dit aussi, dans ce cas, que la clef est secrète (ou bien privée).

-Cryptosystème asymétrique

Si « e » est publique et «d » est privée, on dit que le système (ou la clef) est asymétrique ou encore que c'est un système à clef publique. Dans un tel système asymétrique, le calcul de e (resp. d) en fonction de d (resp. e) doit être infaisable pratiquement.

1.1. Aperçu historique

1.1.1. Le chiffrement de César

Il est considéré comme l'un des premiers crypto-systèmes à clef secrète. La clef **K**correspond à un chiffre entre 0 et 25. Pour chiffrer un message, chaque lettre de cedernier

est décalée de*K* positions (décalage dans l'ordre alphabétique). Déchiffrer le message revient à décaler dans le sens inverse.
En pondérant les lettres de l'alphabet par des coefficients (de 0 à 25), ces deux opérations peuvent être vue comme une addition dans le groupe (Z/26Z, +). Le chiffrement revient à additionner le coefficient de la lettre avec *K*. On déchiffre en ajoutant 26 − *K*, toujours dans Z/26Z.

1.1.2Transposition

Construire des anagrammes en changeant l'ordre des lettres. Connue depuis l'Antiquité (scytale des Spartes) L'ordonnancement des lettres doit suivre un système rigoureux sur lequel d'expéditeur et l'envoyeur ce sont préalablement entendus.

1.1.3. Le chiffrement de Vigenère

Le chiffre de Vigenère est un système de chiffrement poly-alphabétique, c'est un chiffrement par substitution, mais une même lettre du message clair peut, suivant sa position dans celui-ci, être remplacée par des lettres différentes, contrairement à un système de chiffrement mono-alphabétique comme le chiffre de César (qu'il utilise cependant comme composant). Cette méthode résiste ainsi à l'analyse de fréquences, ce qui est un avantage décisif sur les chiffrements mono-alphabétiques.

1.2. Les règles qui assurent la relation complexe entre (M, K, C).

Confusion et diffusionc'est rendre la relation aussi complexe que possible entre M, C, K
M : message

C : cryptogramme

K : clé

1.2.1. Confusion

La **confusion** cache les relations entre le texte clair et le texte chiffré pour éviter les attaques par analyse statistique un changement d'un seul bit dans le texte clair doit affecter un grand nombre de bits (tous) du texte chiffré. C'est l'effet avalanche: petite modification, grand impact.

1.2.2. Diffusion

La **diffusion** disperse la redondance du texte clair dans le texte chiffré, **par exemple** deux lettres doublées ne doivent pas rester côte à côte après cryptage

2. La Cryptanalyse

La cryptanalyse est l'ensemble des techniques permettant à une personne non autorisée de trouver le contenu d'un message.

2.1. Principes de la cryptanalyse:

Les méthodes et outils permettant de découvrir le maximum d'informations secrètes appartenant au domaine de la cryptographie, Clés, textes en clair, langues utilisées, algorithmes de chiffrement.La cryptanalyse cherche à « casser » du code par une combinaison :

- de raisonnement analytique,
- d'application d'outils mathématiques,
- de recherche de modèle,
- de patience, de détermination et aussi de chance

2.1.1. Analyse des fréquences

Chaque langue possède un profil particulier en ce qui concerne la fréquence des lettres, des digrammes, trigrammes, etc.

-Ceci peut être utilisé pour monter une attaque à texte crypté seulement (attaque distinctive, décryptage, récupération de la clé).

-contre les substitutions monoalphabétiques ou de poly grammes.

-On peut contrer partiellement cette attaque en réduisant la redondance du message (p.ex. compression).

2.1.2. L'attaque à l'aide de l'analyse statistique (Statisticalanalysisattack).

Le cryptographe possède des informations sur les statistiques du message clair (fréquences des lettres ou des séquences de lettres).Exemple l'attaque sur le chiffrement par bloc Vigenère (substitution poly-alphabétique), selon la taille de la clé, les statistiques peuvent être utilisées sur les lettres qui sont chiffrées de la même façon plus clairement si on a une clé de taille 3 :

- ✓ Les lettres du texte en clair qui sont aux positions: 0, 3, 6, Sont chiffrées de la même façon: même décalage.
- ✓ Les lettres du texte en clair qui sont aux positions: 1, 4, 7, Sont chiffrées de la même façon: même décalage.
- ✓ Les lettres du texte en clair qui sont aux positions: 2, 5, 8, Sont chiffrées de la même façon: même décalage.

2.1.3.L'attaque en force (Brute force attack, Exhaustive key search attack).

Le cryptographe essaie toutes les combinaisons de clés possibles jusqu'à l'obtention du texte clair.

2.1.4. L'attaque à l'aide de texte chiffré seulement (Ciphertext-onlyattack).

Le cryptographe dispose du message chiffré par l'algorithme et fait des hypothèses sur le texte clair (expressions, mots, sens du message…)

2.1.5. L'attaque à l'aide de texte clair (Known--plaintext attack) ou de texte clair choisi (Chosen--plaintext attack) :

Le cryptographe dispose des messages ou parties de message clairs et de leur version chiffrée et tente de résoudre les mécanismes de chiffrement de l'algorithme.

II. Modélisation du projet

1. langage utilisé « java »

Java est un langage de programmation orienté objet et un environnement d'éxecution, développé par Sun Microsystems. Il fut présenté officiellement en 1995. Le Java était à la base un langage pour Internet, pour pouvoir rendre plus dynamiques les pages (tout comme le JavaScript aujourd'hui). Mais le Java a beaucoup évolué et est devenu un langage de programmation très puissant permettant de presque tout faire. Java et sa JVM permettent au programmeur d'avoir un très haut niveau d'abstraction par rapport à la machine. Il n'aura donc pas besoin de s'occuper de la compatibilité matérielle. Java met à la disposition du développeur une API très riche lui permettant de faire de très nombreuses choses, Ceci est un énorme avantage, cela augmente grandement la productivité de développement. Java est complètement orienté objet. Java permet de développer des applications d'une façon orientée objet et permet d'avoir une application bien structurée, modulable, maintenable beaucoup plus facilement et efficace. Cela augmente une fois de plus la productivité.

Donc en résumé les avantages de Java:

- Portabilité excellente
- Langage puissant
- Langage orienté objet
- Langage de haut niveau
- JDK très riche
- Nombreuses librairies tierces
- Très grande productivité

- Applications plus sûres et stables
- Nombreuses implémentations, JVM et compilateurs, libres ou non
- IDE de très bonne qualité et libres : Eclipse et Netbeans par exemple
- Supporté par de nombreuses entreprises telles que Sun ou encore IBM et des projets comme Apache

2. Attaque sur notre boite noire

3. La boite à outil

3.1. L'interface graphique

3.2. Les fonctions principales (avec le code et les explications)

3.2.1. la fonctionnalité du modulo:

Elle consiste à chercher le plus grand byte contenu dans un message.

Dans cette fonctionnalité on donne en entrée un fichier texte d'une longueur donnée et on aura en sortie le byte le plus grand contenu dans ce fichier et cette nous indiquera la valeur du modulo avec le quel le cryptosystème peut bien travailler c'est à dire savoir si notre notre cryptosysteme travaille avec un modulo caché. Comme le chiffrement de cesar, vigenere.

c'est la classe "taille" dans notre programme.

3.2.2. la fonctionnalité de substitution:

elle consiste à remplacer un caractère donné dans le message en clair par un autre caractère. on donne en entrée un fichier texte et un caractère qu'on veut remplacer dans ce fichier avec le caractère par le quel on désire le remplacer on aura en sortie un autre fichier texte dans le quel on a fait la substitution . et parmi les chiffrement qui utilisent la substitution on site:

le chiffrement par substitution mono-alphabétique: Dans le message clair , on remplace chaque lettre par une lettre différente.

c'est la classe "Substitue" dans notre programme.

3.2.3.chiffrement et déchiffrement par blocs (XOR):

elle prend en entrée un fichier de longueur taille et une clé de taille t, en suite on divise le fichier en blocs de taille t et on fait le XOR entre chaque bloc et la clé donnée en entrée et on aura en sortie un autre fichier qui est le chiffré(si on donne en entrée un message clair) ou le clair(si on donne en entrée un message crypté) du fichier qu'on a donné en entrée.

c'est la classe "ChiffDechiffParBloc" dans notre programme.

3.2.4.Calcul de la taille de la clé:

l'idée c'est d'analyser les séquences de caractères répétés dans un code donné et de trouver les distances entre ces blocs répétés, puis une fois les distances entre les bloques similaires ont été calculées on calcule les diviseurs de ces dernières et on cherche le pgcd de ces distances qui correspond à la taille de la clé utilisée.

Donc on donne en entrée un message d'une longueur donnée et une taille pour les blocs identiques à chercher. En sortie elle retourne la taille de la clé.

c'est la classe "CalculeTaille" dans notre programme.

3.2.5.Calcul de la fréquence maximale dans un fichier à une position donnée:

On cherche caractère le plus répété qui apparait à une position donnée p.

et ceci se fera en divisant le texte en bloc de taille T puis chercher le caractère le plus fréquent à une position p.

Elle prend en entrée un fichier texte, la taille des blocs et la position pour la quelle on désire chercher la fréquence maximale. et en sortie elle retourne la fréquence maximale correspondante.

c'est la classe "CalculFreqMaxPos" dans notre programme.

3.2.6. La substitution par position dans un bloc:

le principe c'est de découper un fichier texte en blocs de taille T et de chercher le caractère qu'il ya à la position i dans le bloc et de vérifier s'il est égale au caractère que l'on veut remplacer puis de le remplacer par le caractère voulu.

Elle prend en entrée un fichier texte, une taille des blocs, une position i, le caractère que l 'on veut remplacer et le caractère par le quel on désire le remplacer. en sortie on aura un fichier texte dans le quel la substitution a été faite.

c'est la classe "SubstitutionParPositionDansLeBloc" dans notre programme.

3.2.7. Conversion de l'Unicode au ASCII UTF-8 (trans)

Le cryptanalyse utilise cette fonction car ils existent différents algorithmes chiffrement qui utilise la conversion en Unicode (exemple les mots de passe) , alors le cryptanalyse a besoin de faire la transformation inverse et tester s'il y a des transformations ou non pour trouver le message clair.

3.2.8. La comparaison des ensembles des caractères de deux fichiers (`FileComparaison`)

Le cryptanalyse utilise cette fonction dans le but à partir de deux fichiers.txt (deux chiffrés déférents), il vérifie est ce qu'ils ont les mêmes caractères (l'utilité de cette fonction est de tester est ce qu'on reste dans le même domaine des caractères (exemple : si on fait un sur-chiffrement d'un chiffré C1,).

3.2.9. Recherche les lignes différentes entre deux fichiers (Comparaison_Lignes)

Elle prend en paramètre deux fichiers et elle donne :

1) Les numéros des lignes et les lignes qui sont présentent dans le fichier1 mais absentent dans le fichier

2) Les numéros des lignes et les lignes commune au fichier1 et au fichier 2 .

3.2.10. L'analyse de fréquences des N-grammes (Ngramme)

Utilité de cette fonctionnalité est de faire l'analyse de fréquences des N-grammes selon le choix de l'utilisateur en mode **: glissant ou non glissant** et on peut effectuer l'analyse selon le choix soit tous **les caractères de texte ou juste les lettre** ou **juste les numéros** ou pour **les lettres et les numéros ensemble**.

C'est une méthode de cryptanalyse consistant à examiner la fréquence des N-grammes employés dans un message chiffré. Cette méthode est utilisée pour décoder des messages chiffrés par substitution.

Un petit exemple :pour N=2

Pour une chaine de caractère ABCDEF, l'analyse des bigrammes glissants donnera 1 fois AB, 1 fois BC, 1 fois CD, 1 fois DE et 1 fois EF

3.2.11. Longueurs des mots (`motlongeur`)

Utilité de cette fonctionnalité et d'effectuer L'analyse des fréquences des longueurs des mots pour un texte donné

3.2.12. Indice de coïncidence (`Coincidence`)

L'indice de coïncidence est une technique de cryptanalyse elle permet de savoir si un texte français a été chiffré avec un chiffre mono-alphabétique ou un chiffre poly-alphabétique en étudiant la probabilité de répétition des lettres du message chiffré (exemple : En français, l'indice de coïncidence vaut environ 0,0778) .

Avec un IC aussi proche du français(0,0778), on peut faire l'hypothèse que ce cryptogramme est le résultat d'un chiffrement mono-alphabétique.

Avec l'IC aussi différent du français (0.0778), on peut déduire que le chiffre utilisé est poly-alphabétique. Il sera donc inutile d'étudier les bigrammes de ce texte.

3.2.13. La longueur de la clé probable avec l'indice de coïncidence (casser le chiffrement de Vigenère) `LongeurCle`

Utilité de cette fonctionnalité c'est lors de la vérification automatique d'un déchiffrement, en particulier lors d'une recherche exhaustive de toutes les clés(pour un chiffré d'un texte francais). Les clés incorrectes fournissent un indice très proche de l'indice de données aléatoires uniformément distribuées(0,038). Une clé donnant un indice plus élevé a donc des chances d'être la bonne.

Exemple de fonctionnement :

Soit le message suivant, chiffré avec Vigenère

PERTQ UDCDJ XESCW MPNLV MIQDI ZTQFV XAKLR PICCP QSHZY DNCPW EAJWS ZGCLM QNRDE OHCGE ZTQZY HELEW AUQFR OICWH QMYRR UFGBY QSEPV NEQCS EEQWE EAGDS ZDCWE OHYDW QERLM FTCCQ UNCPP QSKPY FEQOI OHGPR EERWI EFSDM XSYGE UELEH USNLV GPMFV EIVXS USJPW HIEYS NLCDW MCRTZ MICYX MNMFZ QASLZ QCJPY DSTTK ZEPZR ECMYW OICYG UESIU GIRCE UTYTI ZTJPW HIEYI ETYYH USOFI XESCW HOGDM ZSNLV QSQPY JSCAV QSQLM QNRLP QSRLM XLCCG AMKPG QLYLY DAGEH GERCI RAGEI ZNMGI YBPP

On va considérer les sous-chaînes obtenues en prenant les lettres à intervalle donné:

Intervalle de 1: PERTQ UDCDJ XESCW MPNLV ... (texte original)

Intervalle de 2: PRQDD XSWPL ... et ETUCJ ECMNV ...

Intervalle de 3: PTDJS MLIIQ ... , EQCXC PVQZF... et RUDEW NMDTV

... .

On calcule ensuite les IC pour toutes ces sous-chaînes.

Intervalle Indice de coïncidence

1=> 0.0456107

2=> 0.0476954, 0.0443098

3=> 0.044249, 0.0494469, 0.0426771

4=> 0.0465839, 0.0453894, 0.0449116, 0.0425227

5=> 0.0799704, 0.0925583, 0.0836727, 0.0795282, 0.0684932

6=> 0.0512956, 0.0407192, 0.0371585, 0.0382514, 0.0661202, 0.0431694

On remarque que quand l'intervalle est de 5, l'IC correspond plus ou moins avec l'IC caractéristique du français (en tout cas, c'est cette ligne qui s'approche le plus de 0.074, les autres lignes étant plutôt proches de 0.038). La longueur de la clef utilisée est donc probablement 5

4. Gestionnaire de version

4.1. Présentation du gestionnaire de version choisi

4.2. Description des différentes étapes de l'implémentation

4.3.Le mode d'emploie

5. Description du plan de travail

5.1. La répartition des taches

Pour atteindre les objectifs fixés, nous développons un plan d'action. Notre travail de groupe peut se résumer en deux étapes, le premier objectif c'est de casser la boite noire et le deuxième consiste en une étude des déférentes fonctionnalités qu'un cryptanalyste peut utiliser pour casser n'importe qu'elle crypto-système et les implémenter sou forme d'une boite à outil avec une interface graphique et un guide d'utilisation de cette boite.

Le travail est partagé comme suis :

1. Etudier et faire des recherches sur les déférentes attaques classiques et s'inspirer pour trouver les différentes fonctionnalités pour la boite à outil et les implémenter, et rédiger le rapport .Cette tâche est effectué par Saidani Meriem

2. Implémenter l’interface graphique de la boite à outil et intégré les fonctionnalités que l’autre membre de groupe ont implémentés en java. Cette tâche est effectuée par Mamouni Walid
3. Etudier la boite noire en utilisant tous ses failles et s’inspirer des attaque classique qu’on a déjà cité dans la partie de conception, tel qu’on dispose d'un certain nombre de triplets (clair,clé,chiffré). Cette tâche est effectuée par BAOUR Tarik.
4. Etudier et chercher des fonctionnalités pour la boite à outil et les implémenter en effectuant des recherches en parallèle sur le rapport. Cette tâche est effectuée par HaroucheGhania.
5. Installer un logiciel qui permet de travailler à distance, et implémenter des fonctionnalités pour la boite à outil. Cette tâche est effectué par Soualah Youba.

5.2. Problèmes rencontrés .

5.3. Les aides externes.

Conclusion