Cahier Des Charges

AYOUB Pierre - BASKEVITCH Claire - BESSAC Tristan -
CAUMES Clément - DELAUNAY Damien - DOUDOUH Yassin

Stéganographie & Stéganalyse

Mercredi 14 Mars 2018

1 Préambule

1.1 Définition des termes du sujet

La stéganographie est l'art de la dissimulation, appliquée en informatique en cachant des données dans d'autres données. Cette dissimulation se fait généralement au sein de fichiers multimédias. La stéganographie se différencie de la cryptographie, qui correspond à chiffrer un message afin qu'il soit illisible par une personne différente de l'émetteur et du destinataire. En effet, un message chiffré en cryptographie sera visible par tous mais illisible, tandis qu'un message caché dans un fichier f en stéganographie ne sera vu que si un inconnu sait que f contient un message et connaît l'algorithme pour l'interpréter.

La stéganalyse, quant à elle, est la recherche de données cachées dans des fichiers suspects. Si ces données sont identifiées, il faut ensuite réussir à les extraire pour les lire. Il s'agit donc de la méthode inverse à la stéganographie.

1.2 Historique

La stéganographie est une méthode très ancienne dont la première référence à cette utilisation date du premier siècle avant Jésus-Christ. Elle apparaît dans un récit écrit par Hérodote qui raconte comment deux citoyens communiquaient secrètement : le premier citoyen rasait la tête de son esclave et lui écrivait un message sur son crâne. Ensuite, il fallait attendre que les cheveux de l'esclave repousse puis envoyer ce dernier chez le deuxième citoyen. Ce citoyen devait de nouveau raser la tête de l'esclave pour découvrir le message qui lui était destiné. Une autre utilisation de la stéganographie consistait à utiliser de l'encre, invisible à l'oeil nu, mais qui était révélée à la chaleur.

Avec l'émergence de l'informatique, les techniques de stéganographie se sont renouvelées. En effet, il est désormais possible de cacher des données dans d'autres données. Cette multiplicité de techniques stéganographiques, grâce à l'informatique, montre l'étendue de cette application dans tous les domaines. Par exemple, la stéganographie moderne a été utilisée dans des communications terroristes (transmission de messages) ou dans les signatures de fichiers multimedia (tatouage numérique) afin de protéger les droits d'auteurs.

2 Conducteurs du projet

2.1 But du projet

Le but du projet est de réaliser un logiciel de stéganographie permettant à des personnes lambdas de communiquer sans que l'on soupçonne que leurs communications soient en réalité compromettantes.

Le but de l'application est de permettre à un utilisateur U_1 d'envoyer des données cachées à un autre utilisateur U_2. Ce deuxième utilisateur devra pouvoir interpréter ces données en utilisant la même application que U_1.

2.2 Motivation du projet

La motivation du projet est venue par notre envie de la majorité des membres de ce groupe de projet d'obtenir le master SeCReTs. En effet, nous voulions tous réaliser un projet en rapport à la cryptographie et c'est donc pour cela que nous nous sommes réunis afin de réaliser ce type de projet.

3 Contraintes du projet

3.1 Calendrier

Le Calendrier est imposé et suit les étapes suivantes :

- Le Cahier des Charges doit être remis le 14 mars.
- Le Cahier des Spécifications est à remettre le 18 avril.
- La remise du produit au client est le 25 mai.
- La présentation du produit au client sera le 1 juin 2018.

3.2 Contraintes imposées

Plusieurs contraintes sont imposées par l'utilisateur :

- l'application aura une interface graphique
- le logiciel proposera également une interface en ligne de commande

4 Exigences fonctionnelles

4.1 Portée du produit

L'application sera utilisée par des utilisateurs qui pourront tous faire les mêmes actions : dissimuler des données dans un fichier image, son et vidéo (partie stéganographie). Il pourra également savoir si un fichier contient des données cachées et les extraire (stéganalyse). Cela permettra à un utilisateur d'insérer des données cachées dans un fichier, afin de l'envoyer à un autre utilisateur. Ce dernier va ensuite faire la tâche inverse : extraire ces données cachées.

4.2 Exigences du client

L'application doit respectée deux exigences pour le client : elle doit permettre à un utilisateur ne connaissant pas la stéganographie de pouvoir facilement utiliser toutes les fonctionnalités de l'application grâce à une interface graphique. De plus, une interface au terminal doit être également proposée pour les utilisateurs savant manipuler le terminal. En effet, ces utilisateurs pourront réaliser les mêmes fonctionnalités qu'avec l'interface graphique.

5 Exigences non fonctionnelles

5.1 Apparence et Perception

Le logiciel doit permettre à n'importe quel utilisateur de pouvoir cacher ses données dans des fichiers. En effet, la facilité d'utilisation de l'application sera ciblée pour le développement. Il sera facile de naviguer entre les deux menus : si l'utilisateur veut cacher ses données ou s'il veut découvrir les données cachées dans un fichier. Il pourra naviguer dans son système de gestion de fichiers afin de choisir quelles données cachées et quel fichier contiendra ces données cachées.

5.2 Performance

L'application devra être rapide pour l'utilisateur qui s'en sert. Bien entendu, la stéganographie sur certains fichiers lourds (tels que des fichiers de type video) rendra l'exécution plus lente mais elle ne devra pas être trop importante pour l'utilisateur.

5.3 Exigences culturelles, politiques et légales

L'application de stéganographie vise des clients en France. Il faut donc respecter les lois françaises concernant l'utilisation de cette application. La loi du 21 juin 2004, pour la confiance dans l'économie numérique, définit cette application comme moyen de cryptologie car elle vise à transformer des données pour garantir la sécurité de la transmission de celles-ci. Par ailleurs, l'article 30 de cette même loi oblige la déclaration de l'application si cette dernière est importée et/ou exportée.

6 Modules du produit

6.1 Organigramme

6.2 Algorithmes des fonctionnalités

L'application permettra de cacher des données dans des fichiers de différents formats (image, son, video). Plusieurs algorithmes seront utilisés dans notre logiciel.

6.2.1 Algorithme LSB (Least Significant Bit)

L'algorithme LSB permet de cacher des bits dans des octets tel qu'ils seront invisibles pour l'Homme. Il permet de cacher des données dans un fichier sans en altérer sa taille. Dans le cas de la stéganographie sur image, chaque pixel d'une image correspond à un triplet de nombres : R,G,B qui correspondent aux composantes de couleurs Rouge-Vert-Bleu de 0 à 255. Le but de cet algorithme est donc de cacher des bits dans cette image. Pour se faire, nous allons remplacer les 2 bits de poids faibles de chaque composante des pixels de l'image. En effet, à l'oeil nu, l'homme ne discernera jamais le changement minime de composante. Prenons un exemple de couleur C_1 dont le triplet est $(219, 27, 91)$.

R : $219_{10} = 11011011_2$ G : $27_{10} = 11011_2$ B : $91_{10} = 1011011_2$

Imaginons que la donnée à cacher dans le fichier composé de cet unique pixel de couleur C_1 correspond à la suite de bits $B = 000000_2$ et Ce qui donne une toute autre couleur C_2 en changeant les 2 bits de poids faibles de chaque composantes du pixel :

R : $216_{10} = 11011011_2$ G : $24_{10} = 11000_2$ B : $88_{10} = 1011000_2$

Voici ici les deux couleurs C_1 et C_2, montrant ainsi qu'un humain ne pourra jamais détecter un changement de bit :

FIGURE 1 – Couleur C_1

FIGURE 2 – Couleur C_2

Cet algorithme peut également s'appliquer à d'autres types de fichiers : pour le son, il est possible de modifier très peu les fréquences sonores sans en altérer le bruit ; pour la video, un fichier video est composé de frames (images de la vidéo) et il est donc possible de manipuler les données pour pouvoir en cacher d'autres.

Pour la partie réception du fichier, il faut savoir si ce fichier a été utilisé pour un message caché et connaître combien de bits sont cachés. Il faudra donc calculer la taille maximale du message à cacher qui sera une puissance de 2. En effet, en fonction de la taille du fichier, un certain nombre de bits sera réservé pour connaître la taille des données à cacher. En fonction de ces informations, la suite de bits cachée sera donc formée.

6.2.2 Algorithme EOF (End Of File)

Chaque format de fichier a une mise en forme unique permettant de décrire facilement n'importe quelles données. L'entête du fichier va contenir sa signature (Magic Number) ainsi que plusieurs octets décrivant ce fichier, puis, le réel contenu du fichier, visible par l'utilisateur grâce à un éditeur. Cet éditeur de fichiers va donc lire les données contenues dans le fichier en les interprétant.

Pour que l'éditeur sache quand la lecture doit s'arrêter, le fichier va contenir, à la toute fin, un octet représentant la fin du fichier (EOF). Si des données existent après ce EOF, elles ne seront pas interprétées par l'éditeur. L'algorithme EOF est un algorithme très utilisé dans la stéganographie pour cacher des données : il consiste à écrire une suite de bits, représentant les données à cacher, dans le fichier où l'on va cacher ces dernières.

6.3 Estimations des coûts

7 Autres aspects du projet

7.1 Solutions sur étagère déjà existantes

Plusieurs applications de stéganographie existent déjà. En effet, elles peuvent cacher des données dans différents formats de fichiers image, audio et video. Pourtant, nous avons recensé 23 applications de stéganographie et seulement une seule propose de cacher des données dans des fichiers image, son et video. Tous les autres ne s'occupent que des images.

7.2 Tâches à réaliser pour le développement de l'application

Un manuel d'utilisation sera créé et disponible lors du rendu de l'application au client.

7.3 Améliorations pour les versions futures du projet

La stéganographie se distingue de la cryptographie par le fait que, dans l'un, le message caché est visible par tous si celui-ci est extrait ; tandis que, dans l'autre, le message est transmis en étant chiffré sur un canal non-sûr.

Pour une projection à long terme, dans d'éventuelles versions du logiciel, nous pourrions améliorer l'application en chiffrant les données cachées. En effet, lors de l'interception d'un éventuel fichier cachant des données, il faudra, en plus de les extraire, les déchiffrer : ce qui rend la tâche beaucoup plus longue pour celui qui intercepte le fichier et qui tente de récupérer ces données cachées.

7.4 Choix du langage et de l'interface

8 Conclusion