Stéganographie : Bitmap et GIF

Foud Hind et Patti Philippe

3 avril 2020

Table des matières

1 Introduction

L'objectif du projet est d'implémenter un programme employant la stéganographie.

Dans le cadre du cours de Systèmes, la stéganographie nous permet d'approfondir l'utilisation de la mémoire pour stocker des informations. Nous devons apprendre en détail la structure des fichiers bitmaps et GIF pour pouvoir y cacher des messages. De plus, ce sujet nous permet d'approfondir le thème de la sécurité, un thème qui est fort lié aux systèmes d'exploitations.

Nous avons choisi d'implémenter 2 programmes en C qui cacheront un message à l'intérieur d'un bitmap et d'un GIF. Dans ce rapport, nous expliquerons pour chaque format la raison de notre choix, la technique utilisée, l'avancement du projet et un guide pour utiliser les executables.

2 Stéganographie : Définition

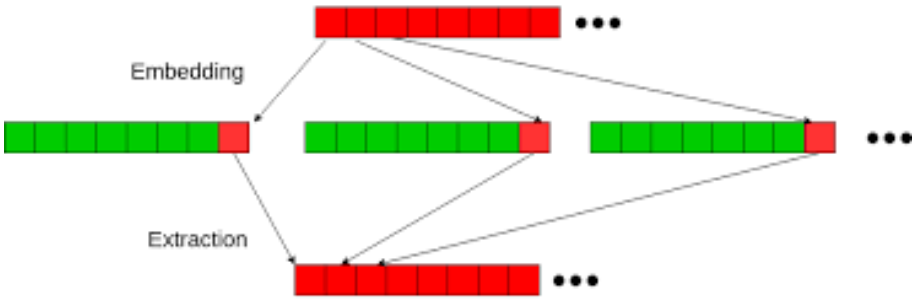
La stéganographie est l'art de la dissimulation : son objet est de faire passer inaperçu un message dans un autre message. Elle se distingue de la cryptographie, « art du secret », qui cherche à rendre un message inintelligible à autre que qui-de-droit.

3 Technique utilisée

Pour cacher un message, nous employons la technique Least Significant Bit (LSB).

Comme vous le voyez sur le schéma ci-dessous, cette technique consiste à caché un bit d'information dans le bit le moins important d'un byte. Pour décripter le message, il suffit de lire le dernier bit de ces bytes et de reformer un message compréhensible.

Pour que le message ait le plus petit impact possible sur le fichier source, il faut employer des bytes qui peuvent perdre un bit d'information. Par exemple, pour une couleur codée en rgb, le changement du dernier bit a un impact très faible, quasi invisible à l'oeil nu.



4 Guide pratique

Pour exécuter le projet, exécuter le makefile à la racine du fichier. Voici quelques options possibles avec le makefile :

```
#NOM : Makefile
#CLASSE : SYSG5
```

```
#OBJET : Steganographie avec Bitmap et GIF
#AUTEUR : Foud Hind et Patti Philippe
#HOWTO : make ; make run_gif; make build_gif; make clean

# make                : execute les demos
# make run            : execute les demos
# make build          : compile les demos
# make clean          : supprime les fichiers generes

# make run_bmp        : execute uniquement la demo pour le format bmp
# make build_bmp      : compile uniquement la demo pour le format bmp
# make clean_bmp      : supprime uniquement les fichiers generes lies au format bmp

# make run_gif        : execute uniquement la demo pour le format GIF
# make build_gif      : compile uniquement la demo pour le format GIF
# make clean_gif      : supprime uniquement les fichiers generes lies au format GIF
```

5 Bitmap

5.1 Pourquoi le bitmap ?

5.2 Structure

5.3 Application du LSB

5.4 Avancement

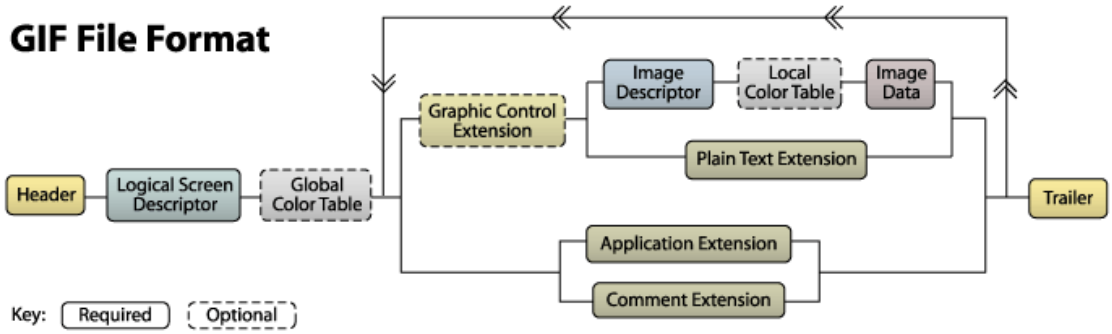
5.5 Guide pratique

6 GIF

6.1 Pourquoi le GIF ?

Le format GIF permet de stocker plusieurs images dans un fichier. Ceci permet de créer des diaporamas, voire des animations si les images sont affichées à un rythme suffisamment soutenu. Chaque image d'une animation peut avoir sa propre palette. Le GIF est un format très utilisé, particulièrement sur les réseaux sociaux. Ce projet peut donc intéressé pas mal de gens.

La structure du format GIF est connue et disponible en ligne. En voici un schéma :



6.2 Application du LSB

Nous avons choisi de cacher des informations dans les couleurs qui se trouvent dans les Local Color Table. Cette section facultative peut revenir devant chaque bloc image data. Si cette section n'existe pas, nous la rajoutons.

Il est possible de calculer la taille maximale du message cachée à l'avance : en comptant le nombre de bloc image data et en le multipliant par le nombre de byte dans la Global Color Table.

6.3 Avancement

6.3.1 Lecture

Le programme peut déja lire un gif en entier, section par section. Les tailles des sections sont indiquées à des endroits différents par sections. La lecture se fait donc différemment en fonction de la section.

De plus, on peut déja savoir le nombre maximal de Local Color Table pour le fichier modifié.

6.3.2 Lecture

Le programme réécrit le gif dans un nouveau fichier, en insérant des Local Color Table, identique à la Global Color Table. Ceci permettra d'y insérer un message. L'insertion du message devrait être évidente, vue qu'elle sera presque identique à la stéganographie dans un fichier bitmap.

Il y a toutefois encore un bug dans la réécriture du fichier gif, probablement à l'écriture des nouvelles Local Color Table.

7 Conclusion