Stéganographie

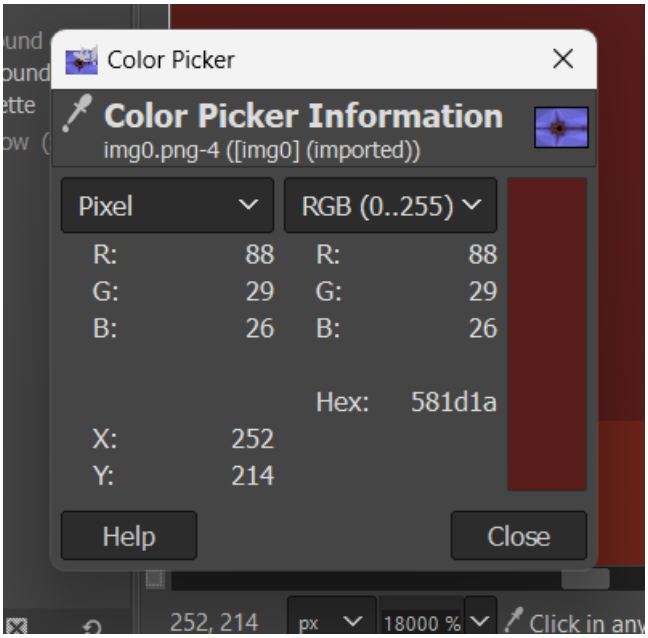
La stéganographie est la dissimulation d'un message. Le but est de passer inaperçu un message dans un autre message. Cependant, ce message dissimulé doit être inintelligible pour tout le monde sauf que la personne à qui le message est destiné. Pour cela, on utilise le changement de couleur des pixels dans une image pour montrer comment la stéganographie fonctionne.

Les pixels sont les petites zones carrées dans une image informatisée. Chaque pixel possède une couleur ou une nuance de gris. Imaginons une image comme un système de coordonnées avec deux axes, X et Y, et chaque pixel se voit attribué une coordonnée (X, Y).

Couleur d'un pixel :

En utilisant de logiciel GIMP on peut trouver des détails sur la couleur d'un pixel.

Dans notre exemple on ouvre l'image « img0.png » dans GIMP et on zoome au plus près possible pour pouvoir voir clairement les pixels de l'image. Si on cherche maintenant notre pixel d'exemple qui est trouvable sous les coordonnées (252, 214) et on clique, avec la fonction de pincette au même temps qu'on appuie sur la touche « shift », sur le pixel, il s'affiche de l'information sur la couleur du pixel et on voit que le code couleur hexadécimal, utilisé en HTML, de notre pixel est 581d1a.

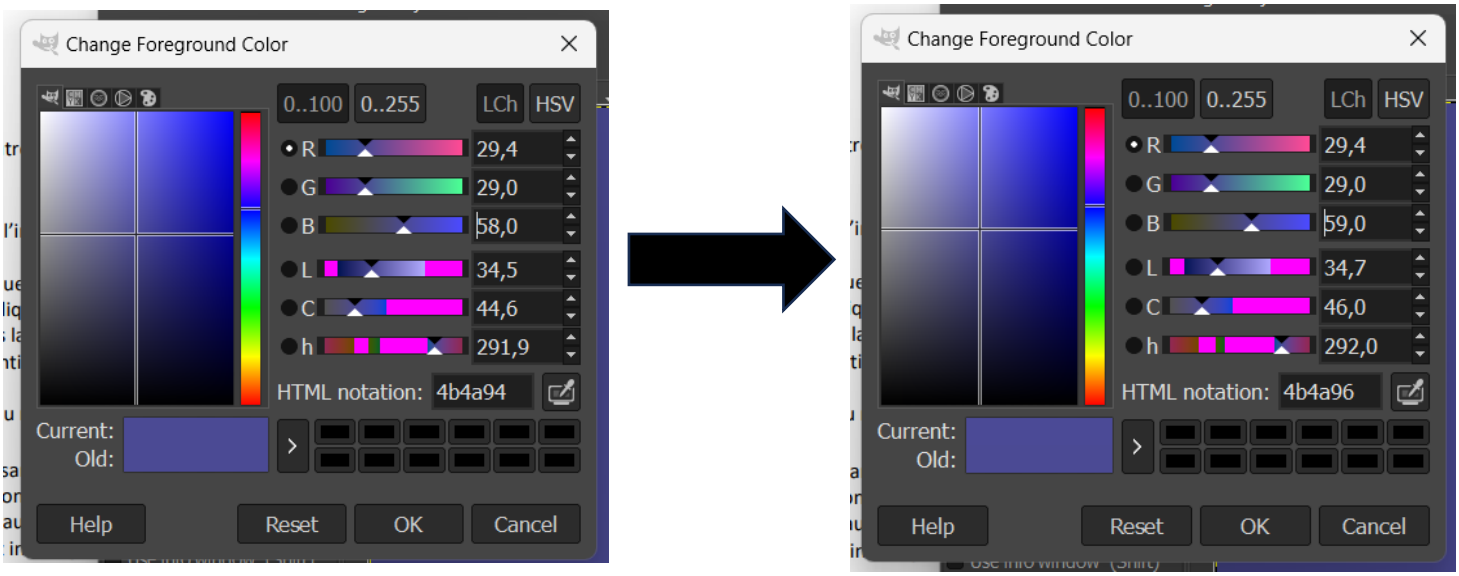


Le code couleur hexadécimal (Hex) de pixel (252, 214) est 581d1a

Description du procédé stéganographie :

Dans la stéganographie on change la couleur d'un pixel seulement un petit peu pour avoir une différente couleur, mais ce change est tellement petit que c'est presqu'á impossible pour l'humain de le voir.

Dans notre exemple on utilise encore la pincette pour vérifier que les deux pixels de coordonnées (0, 0) et (0, 1) ont la même code couleur RGB (Rouge, Vert, Bleue) qui est (75, 74, 149). Notre couleur est affichée à gauche dans la « boîte à outils » et si on clique sur la couleur on ouvre un onglet qui nous permet de la changer. On augmente la couleur bleue (B) par 1 et on clique « ok ».



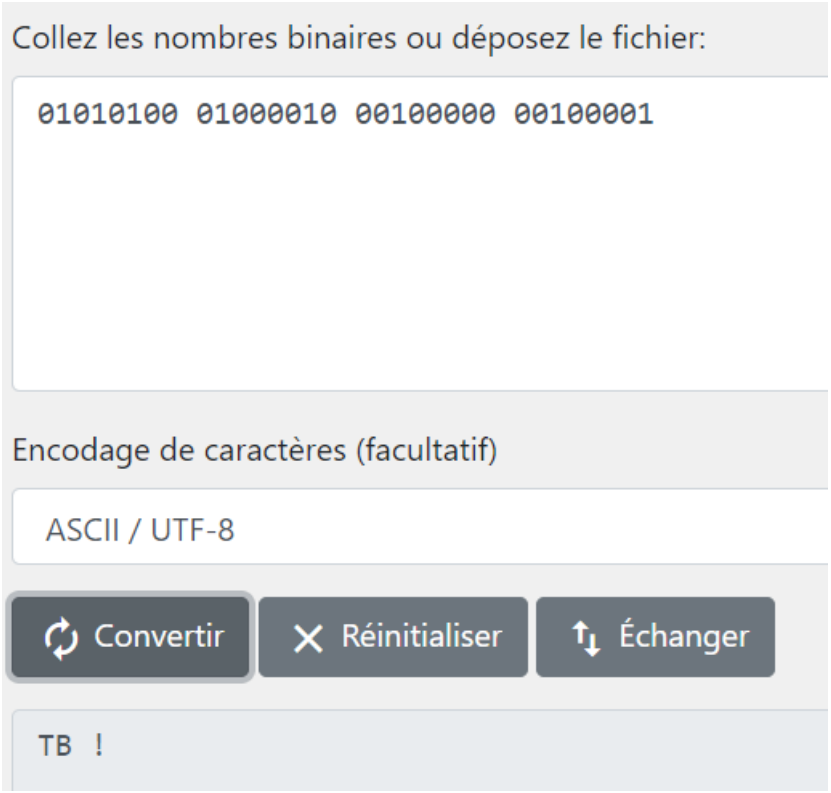
De (29.4, 29, 58,) en (29.4, 29, 59)

Si on compare les deux couleurs on voit que c'est impossible pour nous de les différencier, même si elles sont maintenant différentes. Dans la stéganographie on exploite notre incapacité de différencier des petites différences pour cacher un bit dans un pixel.

Des bits enchainés font un code binaire qui décrit quelque chose, comme une couleur. Si notre bit caché est 0 on modifie la couleur pour que son écriture binaire se termine par 0 et si notre bit caché est 1 on modifie la couleur que son écriture binaire se termine par 1, tous les autres bits restent inchangés. Pour dissimuler un message composé de n bits, on cache chaque bit dans n pixels convenus de l'image.

Retrouver un message :

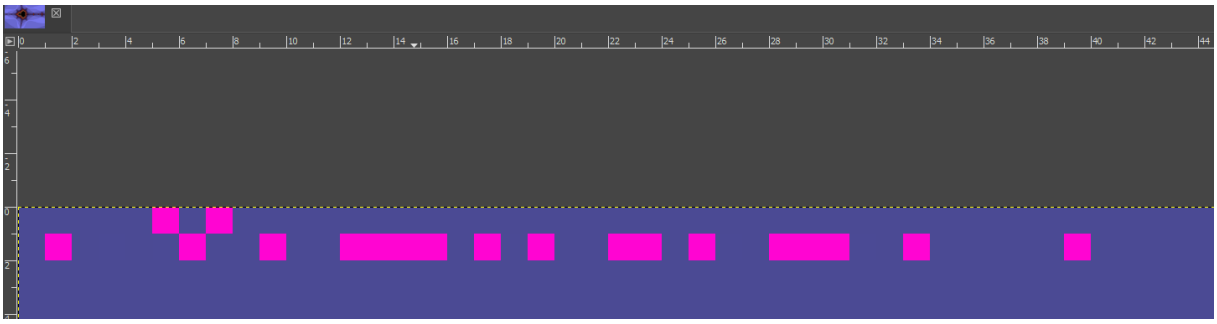
Dans notre exemple on ouvre l'image « stegano-img0.png » en GIMP. Il faut savoir deux choses sur le message dissimulé pour pouvoir le trouver : la longueur et le contenu. Les huit premiers pixels dans la première ligne (donc 0, 0-7) nous indiquent la longueur du message dissimulé. Dépendant de la longueur du message dissimulé, le contenu est indiqué par la deuxième ligne (donc X, 1). Les pixels qui ont B = 148 sont des 0 et les pixels qui ont B = 149 sont des 1. Si on regard sur les premiers huit pixels dans la première ligne on reçois « 00000100 », un code binaire qui peut se traduire en « 4 ». Donc notre message a 4 caractères. Dans le codage ASCII un caractère constitue de 8 bits, ce que veut dire que le message constitue de (4x8 =) 32 bits, donc le message est caché dans les premiers 32 pixels de la deuxième ligne. Si on regard sur ces 32 pixels on reçoit « 01010100 01000010 00100000 00100001 », un code binaire en ASCII qui peut se traduire en « TB ! ».



Traduction en « TB ! »

Dissimuler un message :

Maintenant on sait comment lire un message dissimulé, mais comment peut-on dissimuler un message nous-même ? On fait les mêmes étapes, mais dans l'inverse. Dans cet exemple je choisi le mot « BOSNA ». Si on traduit ce mot en code binaire en ASCII on reçoit « 01000010 01001111 01010011 01001110 01000001 ». Maintenant on a le contenu, mais il manque la longueur pour que le destinataire peut s'assurer de contenu. « BOSNA » a cinq caractères et si on traduit « 5 » en code binaire on reçoit « 00000101 ». Si on utilise le même schéma des couleurs et les mêmes coordonnées, les pixels de la première ligne des coordonnées (0, 0/1/4/6) restent en B = 148, cependant les pixels des coordonnées (0, 2/3/5/7) changent en B = 149. Le même schéma sera appliqué sur les premiers (5x8 =) 40 pixels de la deuxième ligne. Évidement cela marche pour chaque couleur et chaque coordonnée, la seule chose qui est importante est que le destinataire connait le schéma pour pouvoir retracer le message dissimulé par l'envoyeur.



B=148 en bleu et B=149 en rose donnant « 00000101 » et « 01000010 01001111 01010011 01001110 01000001 »

Choix du format de sauvegarde du ficher :

Il existe plusieurs formats de sauvegarde des images dans des fichiers avec des propriétés différentes. Notre image « stegano-img0.png » est en format « png ». Dans GIMP, on export cette image au format « jpg ». Si on ouvre l'image en « jpg » et on essaie de retrouver le message dissimulé, on voit que là où le message était caché, la couleur de tous les pixels est en B=149 à la place de varier entre B=148 et B=149.

Si compare maintenant la taille de ces deux fichiers, on voit que le fichier en « jpg » est plus grand que le fichier en « png ». C'est à cause de la façon de « jpg » de compresser une image. La compression est utile quand on a des images de haute qualité dans lesquelles presqu'à chaque pixel à une autre couleur, comme dans les photographies. Quand il faut définir chaque pixel individuellement, il y a beaucoup de l'information à sauvegarder causant que l'image devient trop grande. Dans la compression en « jpg » par contre, les pixels ne sont pas définis individuellement mais en blocs avec une certaine longueur et chaque bloc possède une couleur. Cela nous permet de sauver des bits et donc de la taille. Dans notre exemple, dans le premier bloc, qui constitue de premiers 8 pixels, il y a 7 pixels avec une couleur de B=148 et un pixel avec B=149. Après la compression tout le bloc est en couleur de B=149, donc la taille des pixels augmentée par 1 bit. Cela est répété pour chaque bloc dans l'image. Notre image n'est pas complexe ce que veut dire qu'on ne profit pas de cette compression. On augment la taille plus qu'on sauve ce qui est la raison pourquoi le format « jpg », qui est normalement plus petite, est maintenant plus grand que le format « png ». Des autres formats sans perte comme PNG sont BMP et RAW. JPG est un format avec perte.

Vers l'infini et au-delà !

Il y a cinq types de la stéganographie : Image, comme dans notre exemple, mais aussi Texte, Vidéo, Audio et Réseau. Ce que veut dire que la stéganographie peut être utilisée partout. Un exemple est la distribution de l'information. Si on ne veut pas envoyer de l'information à chaque personne individuellement parce que c'est fatigant ou pas sécure, on peut dissimuler cette l'information dans un fichier qui est facile à accéder. Comme tout le monde a accès sur ce fichier, certaines personnes peuvent, si elles connaissent la méthode, facilement accéder l'information dissimulée dans le fichier.

S'il vous plait dites moi si je fais des erreurs grammaticales répétitives.