Projet de stéganographie

Rappel du contexte

Pour tout contact rapide : alt.g2-ex9c5kp@monmail.fr.nf

Vous faites partie d'une RedTeam chargé de tester la sécurité d'un serveur d'image hosté dans un hidden service (http://taorwbjrx5ugswsz.onion)
Pour vous connecter au site, vous pouvez télécharger le tor browser bundle https://www.torproject.org/download/

Un de vos collègue a trouvé une vulnérabilité et semble pouvoir exécuter du code sur le serveur de contenu.
La BlueTeam a détecté son reverse-shell et filtre désormais les contenus, seul des images jpg peuvent transiter entre ce serveur et vous.

La vulnérabilité trouvée par votre collègue est encore présente mais ne va pas tarder à se faire corriger.

Vous avez jusqu'à dimanche minuit pour développer un remote-shell stéganographié.

Si l'outil bypass des outils/types standards de stéganalyse, vous aurez plus de points que la moyenne !

Générez une clef gpg de signature, exportez la partie publique et saisissez la dans le champ de saisie texte du moodle.

Archivez les sources de votre programme et déposez les sur le moodle.

Composition du programme

Deux binaires:

- un pour le serveur qui permettra d'exécuter les commandes des fichiers png et d'écrire le résultat dedans.
- un pour le client qui permet d'écrire, et de lire le résultat des png déjà posté sur le serveur

Trois fonctions dans le premier binaire :

- read_stegano : Fonction de lecture de l'image.
- write_stegano : Fonction d'écriture dans l'image.
- bash : Fonction d'exécution de ce qui est marqué dans l'image

- add_png : Fonction qui modifie l'extension du fichier pour qu'il soit en .png et non plus en .img
- remove_png : Fonction qui enleve le .png pour mettre un .img

Deux dans le second :

- read_stegano : Fonction de lecture de l'image.
- write_stegano : Fonction d'écriture dans l'image.
- encrypt : Fonction de chiffrement -- En cours de rédaction
- decrypt : Fonction de déchiffrement -- En cours de rédaction

Utilisation du premier binaire:

```
/path/BinServerUpload <file>
```

Utilisation du deuxième binaire :

```
/path/BinLectureEcriture <file> [option [-w <message>]]
```

Si un paramètre renseigné (i.e 'file') automatiquement le binaire lit le fichier cible. Si un second paramètre est renseigné et s'il est égal à '-w' alors on écrit ce message dans l'image.

Méthode pour compiler

Il faut se mettre dans le dossier et, avec la commande de rust, taper :

```
cargo build
```

Le binaire se trouve dans target/debug/

Méthode de stéganographie utilisée

Pour changer du TP et de la stéganographie dans le RGB, j'ai opté pour le mettre dans l'alpha des images.
Ce n'est sûrement pas la meilleure façon, mais cela permettait de voir autre chose que durant le TP tout en gardant la même idée de modification des pixels de l'image.