Heroctf v3 - Ai Authentification

On nous demandait de contourner un système d'authentification sans mot de passe sur un site web.

Très vite, on se rend compte que le système utilise la reconnaissance faciale pour reconnaître un utilisateur. On met de côté la photo du CEO présente sur le site.

Aussi, en fouillant un peu dans le javascript de la page, on tombe sur cette fonction.

```
function send_to_auth()
{
    // ...
    var img = new Image();
    img.src = canvas.toDataURL();
    socket.emit('authentication', {data: img.src});
}
```

Sans chercher à le comprendre de fond en comble, on voit que la dernière ligne envoi ce que l'on peut présumer être une image. Celle ci est obtenue grâce à la méthode `toDataURL()`.

`canva` contient l'image de notre webcam à l'instant et `toDataURL()` nous donne son encodage Base64 d'après la documentation officielle.

Rien ne nous empêche visiblement d'éxecuter le même appel à `socket.emit()` depuis la console du navigateur (`ctrl-shift-j` sous Firefox) en changeant l'image pour celle du CEO.

Il nous faudra cependant l'encoder en base64. (Par exemple, avec https://www.base64-image.de/).

Ce qui nous donne l'appel suivant:

```
socket.emit('authentication', {data: _arrayBufferToBase64("L'IMAGE EN BASE64")});
```

Le flag: `Hero{H3yyyy_Th4t5_pr3tTy_gO0d}`

Heroctf v3 - Easy Assembly

En analysant le code assembleur on trouve que le flag valide l'équation suivante:

```
flag >> 2 == 1337404
```

Le flag est donc `1337404 << 2`

Heroctf v3 - Find Me

On souhaite trouver le lieu où une photo a été prise.

Nous avons tenté deux méthode, d'abord par l'analyse des métadonnées, infructueuse. Puis en utilisant la recherche par image de Google. Nous avons ainsi pu trouver d'autres photos du même endroit, accompagnées de plus d'informations sur le lieu en question.

Heroctfv3 - Holy Abbot

On nous présente le message suivant:

```
Dans son règne, Dans la béatitude
À tout jamais Dans son règne
Dans son royaume Irrévocablement
Dans son royaume Dans la béatitude
Dans son royaume Irrévocablement
Toujours En paradis
```

Le fait que certaines propositions soient répétées plusieurs fois à l'identique semble indiquer qu'à une lettre du message en clair correspond une proposition du message crypté.

Le nom du challenge n'est pas anodin, nous cherchons un homme d'Église qui aurait travaillé dans la cryptographie Une rapide recherche nous présente le travail de Johannes Trithemius à la fin du 15ème siècle.

la solution se trouve dans sa méthode de chiffrement par subsitution Ave Maria.

Heroctfv3 - Nice PDF

Le PDF contenait des caractères encodé dans des whitespaces. Ceux ci étaient donc invisibles dans un viewer classique, même en sélectionnant pour montrer les lettres en blanc sur fond blanc.

En extrayant le texte du PDF avec un outil approprié, on trouve des lettres qui n'ont apparemment rien à faire là entre certains mots. Ces lettres consituent notre flag.

Heroctf v3 - Ping Pong

On dispose d'un fichier dans lequel s'alternent les mots PING et PONG. L'ordre dans lequel ces mots s'enchaînent n'est pas dû au hasard.

On commence par interpréter chaque mot comme un 1 ou un 0 afin de former une série d'octets encodés en binaire, ensuite nous faisons la conversion vers le charset ASCII. Le résultat est le flag recherché.

Heroctf v3 - Pushhhh

En allant sur le repository du HeroCTF v2, on se rend compte qu'il existe deux branches, le flag se trouvait dans le dernier message de commit de la branche *flag*.

Heroctf v3 - PwnQL

On constate que l'administrateur système n'a pas correctement configuré les permissions de ses fichiers. Ainsi, un fichier `.php.inc` peut être accéder via le navigateur. On y trouve l'algorithme qui permet à un utilisateur de se connecter.

On peut voir que la validation du mot de passe se fait ici avec l'opérateur SQL `LIKE`. Ceci consistue une faille de sécurité puisqu'il nous permet d'entré simplement le caractère `%` comme mot de passe. En effet, ce caractère est cponsidéré comme une wildcard par LIKE.

Heroctf v3 - Record

En analysant les entrées DNS du nom de domaine, on trouve un flag dans une entrée TXT.

Heroctf v3 - Russian Doll

On a une archive de plus de 600 dossiers imbriqués les uns dans les autres. Pour éviter de parcourir cette arborescence à la main, on préfère utiliser la commande find(1) pour chercher et afficher le fichier qui contient le flag.

```
find . -name '*flag*' -type f -exec cat {} \;
```

Heroctf v3 - Transfer

On nous donne le hash suivant: `bfa8b1c8b7f6acc867d6984968756cafd0da99060aa6d90dfb0db1a9ef0c96a2`.

On imagine qu'il s'agit du hash d'une transaction Bitcoin; nous nous rendons donc sur https://blockchain.info pour obtenir l'adresse du portefeuille destinataire de la transaction.

C'est notre flag: `Hero{1PQBh6tddet14bYDsSbUiox1YJb5EL185A}`