Fiche technique du projet de la cryptographie

Attestation de réussite sécurisée

Authentification, OTP, Web sécurisé & Stéganographie

Formation : ING3 – CYBERSECURITY
Nature : Etude, réalisation
Année : 2021/2022

Descriptif :

CY TECH vous sollicite pour mettre en œuvre un procédé de diffusion électronique sécurisée d'attestation de réussite/diplômes aux formations qu'elle délivre.
Il faut garantir l'authenticité de l'attestation délivrée de manière électronique sous forme d'image :

- L'image contient une information visible :
    - le nom de la personne recevant l'attestation de réussite ;
    - le nom de la certification réussie ;
    - un QRcode contenant la signature de ces informations ;
- L'image contient une information dissimulée :
    - une information infalsifiable est dissimulée par stéganographie dans l'image. Cette information reprend les informations visibles de l'attestation ainsi que la date de délivrance garantie par un « timestamp » signé par une autorité d'horodatage « www.freetsa.org » par exemple.

L'école conçoit ce procédé en plusieurs composants :

- la partie « SmartPhone » destinée au responsable de la certification, sous forme d'une application pour son téléphone :
    - destinée à créer un OTP, « *One Time Password* » permettant d'identifier le responsable auprès de l'application, afin d'autoriser sa demande de délivrance d'attestation ;
- la partie « Application » :
    - une application destinée à recevoir la demande de délivrance d'attestation et de la traiter :
        - vérification de l'OTP ;
        - récupération du nom, prénom, adresse électronique de l'étudiant, intitulé de la certification et signature de ces informations avec la date de demande ;

- obtention d'un « timestamp », ou estampille temporelle auprès du tiers horodateur ;
- dissimulation de cette estampille par stéganographie dans l'image et intégration de la signature dans le QRcode ;
- envoi par courrier sécurisé de cette image à l'étudiant ;

o un programme permettant d'extraire et de vérifier :
- l'estampille dissimulée dans l'image par stéganopgraphie ;
- la signature codée dans le QRcode.

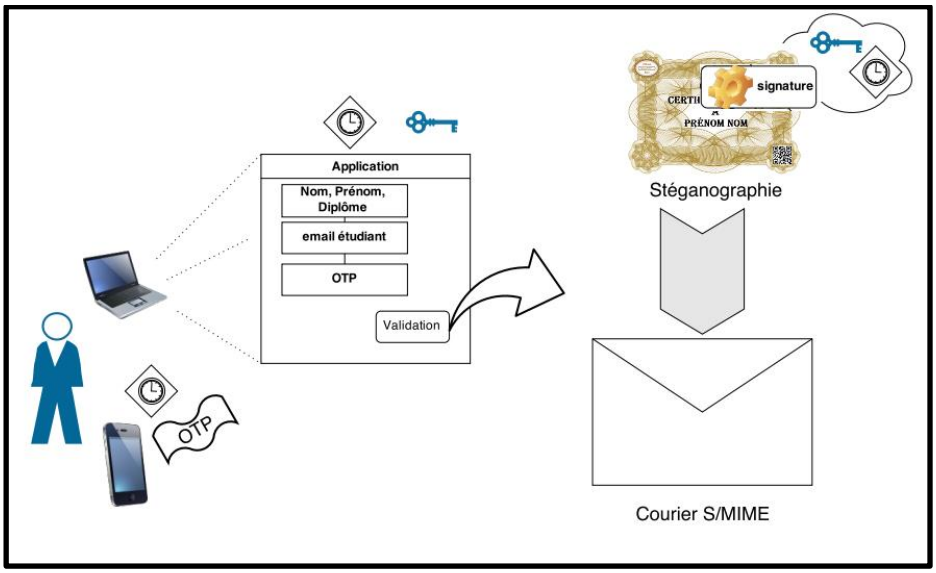

Schéma fonctionnel

CY TECH va se comporter en tant qu'« Autorité de Certification » pour se délivrer des certificats suivant les usages dont elle aura besoin pour réaliser son service.

Les différents programmes à concevoir

- le programme `CreerPass` qui, en fonction de l'heure de son exécution, fournit un OTP, que le client peut ensuite utiliser dans l'application pour autoriser sa demande ;
- le programme `ExtrairePreuve` qui :
  - lit l'image diffusée et récupère le contenu dissimulé par stéganographie :

    ```
    Nom Prénom || Intitulé certification || timestamp
    ```

    Où l'opérateur `||` *désigne la concaténation.*
    Le nom et prénom, ainsi que l'intitulé de la certification devront être complétés afin d'obtenir une chaîne de 64 caractères.
    Le « timestamp » a une taille fixe de n octets, dépendant de la taille de clé utilisée par le service d'horodatage.
  - découpe le contenu en deux parties : informations sur 64 octets et estampille sur *n* octets ;
  - vérifie le « timestamp » par rapport à l'AC d'horodatage ;
  - récupère la partie QRcode de l'image pour récupérer la signature des données ;

- vérifie la signature de la partie informations avec la clé publique de CY TECH ;
- affiche un rapport quant au résultat de cette vérification.

- le programme `CreerAttestation` qui fournit les services suivants :
  - l'accès à la signature par l'AC doit être protégé par l'utilisation d'un secret connu uniquement par la personne habilitée : on utilisera un OTP pour prouver la connaissance de ce secret ;
  - saisie du nom, prénom, intitulé de la certification, adresse électronique de l'étudiant, valeur de l'OTP.
  - vérification de l'OTP ;
  - création de l'image :
    - utilisation d'un fond contenant un tracé infalsifiable
    - intégration d'un QRCode contenant la signature que vous aurez converties au format ASCII ;
    - intégration du texte Nom Prénom et intitulé de la certification.

  - construction du bloc d'informations (Nom Prénom, Intitulé certification)
  - dissimulation par stéganographie du bloc d'information et du « timestamp » dans l'image ;
  - ajout du QRcode contenant la signature de ce bloc avec la clé privée de CY TECH,
  - envoi de l'image modifiée par courrier sécurisé à l'utilisateur.

Travail à réaliser

- créer une AC avec une configuration bien choisie
- créer un certificat permettant la signature ;
- créer un programme construisant un OTP à la manière de celui utilisé par Google ;
- choisir un algorithme de signature et déduire la taille *n* de la signature ;
- écrire l'application chargée de traiter les données soumises par l'utilisateur :
  - calcul OTP et comparaison avec celui fourni à l'application ;
  - construction du bloc d'informations à insérer dans l'image (concaténations et signature) ;
  - insertion de ce bloc par stéganographie et envoi sécurisé de l'image obtenue ;
  - envoi par courrier au format S/MIME, dérivé de PKCS#7.
- créer le programme d'extraction de preuve permettant de vérifier l'attestation.

Pour la création de l'OTP

Le but de l'OTP, « One Time Password » est de dériver de manière unique une valeur depuis un secret partagé entre deux individus sans révéler ce secret partagé.

Pour dériver de manière unique, on va utiliser :

- une valeur qui varie tout le temps : la date EPOCH Unix qui correspond au nombre de secondes écoulées depuis le 1er janvier 1970 ;
- un HMAC utilisant le secret partagé comme clé.

Pour réaliser l'OTP, on utilisera l'algorithme proposé par Google :

Google Authenticator

```
1  function GoogleAuthenticatorCode(string secret)
2  key := base32decode(secret)
3  message := current Unix time ÷ 30
4  hash := HMAC-SHA1(key, message)
5  offset := last nibble of hash
6  //4 bytes starting at the offset
7  truncatedHash := hash[offset..offset+3]
8  //Set the first bit of truncatedHash to zero
9  //remove the most significant bit
10 code := truncatedHash mod 1000000
11 pad code with 0 until length of code is 6
12 return code
```

d'après Wikipedia, https://en.wikipedia.org/wiki/Google_Authenticator

Soit les commandes suivantes :

Donne la date courante considérée en (heures, minutes, secondes et jour, mois, année) en nombre de secondes à partir du 1er janvier 1970 (date de référence d'Unix appelée EPOCH)

```
# date
Thu Apr 4 10:40:34 UTC 2013
# date +%s
1365072036
```

Convertit une date depuis la notation EPOCH vers la notation humaine :

```
# date -d @1365072036
Thu Apr 4 10:40:36 UTC 2013
```

Jeu de test pour l'OTP en utilisant le secret : `fnqurbkuyismvvqo`

```
$ python otp_google.py
date utilisee : 1365072961
OTP : 173475

$ python otp_google.py
```

```
date utilisee : 1365073017
OTP : 968655
```

Pour l'encodage en base32:

```
>>> import base64
>>> base64.b32decode("fnqurbkuyismvvqo".upper())
'+aH\x85T\xc2$\xca\xd6\x0e'
```

Dans le cadre du projet, le secret est partagé entre le logiciel utilisé par l'utilisateur et l'application, et permet à cette application de s'assurer que l'utilisateur est légitime pour construire l'attestation.

Stéganographie

Le programme ci-joint (voir drive) dissimule ou récupère les données en modifiant les bits de poids faible de la composante rouge de l'image.

Vous adapterez ce code à votre programme en utilisant la taille du bloc d'informations + celle du timestamp pour paramétrer la récupération automatique des données.

Envoi de courrier au format S/MIME, PKCS#7

L'envoi de message sécurisé utilise :

- la bibliothèque « smtplib » pour l'envoi effectif du courrier.
- la commande openssl avec l'option smime et le certifcat de CY TECH à utiliser pour réaliser la signature du courrier au format S/MIME :

```
$ cat contenu.txt | openssl smime -signer certif_app.pem -
from 'user1@domaine'-to 'user1@domaine' -subject "Envoi avec
signature" –sign -inkey cle_app.key
–out contenu_courrier.txt
```

Où le fichier « `contenu.txt` » contient l'en-tête suivante suivie du contenu de l'image encodée en base64 :

```
$ cat contenu.txt
Content-Type: image/png
Content-Transfer-Encoding: base64
iVBORw0KGgoAAAANSUhEUgAAAOAAAAEgBAMAAABfnGLSAAAAHlBMVEUAAAAh
Id7/uJfe3t7/AAD/uN4A/97/uEf//wDel0cDUFT6AAAFXUlEQVR42uySgQYD
MRBEh/mh4f3/v1VOVlpXJ71oKXluCYZ3dkevIEcYooEcIUCG490AyUQQQwTo
Y9yFESdh/0roHC8qL3SHa6HzLKz5hhC60LhWBxlCI4gmmL/h+WcooUGrK9X7
Gw6hI8YNyT0hUKuaaGmqpY65bunGtJnOrROnzXRular/em4LC5Q2s7mft/Tv
2Gw2D3bNALVCGAbDjXgAewPxCm8HENYr7P5XWVgHQX4NbVOz5+ivhfKT+kH3
Yluz6KgfXnAUwyi4amEgxWotDGgbl69qEbe2cRQWapuZ0DauFTiAAzitNU2A
...
```

Explications :

- vous codez le contenu de l'image en base64 ;
- vous ajoutez l'en-tête MIME
  « Content-Type: image/png\r\nContent-Transfer-Encoding: base64\r\n\r\n » décrivant ce contenu ;
- vous transférez ce contenu et cette entête à la commande openssl qui réalise la signature et la construction finale du courrier.

un programme Python pour l'envoi du courrier avec une connexion SSL :

```
import smtplib

# message_securise doit contenir le résultat de la commande
openssl
server = smtplib.SMTP_SSL('smtp.domaine',num_port)
#server.set_debuglevel(1)
server.sendmail(adr_exp, adr_dest, message_securise)
server.quit()
```

Travail à soumettre pour l'évaluation :

1. Travail à réaliser par binôme/trinôme.
2. Livrable à rendre le **25 octobre 2021 au plus tard**.
3. Présentation & démonstration.
4. Écrire les différents programmes et contenus demandés.
5. Rédiger un court rapport, au format PDF, pour détailler le fonctionnement de votre programme, avec un exemple d'exécution (notice d'utilisation avec copie d'écran).
6. Livrables : une archive contenant :
   a. le certificat racine de l'AC ;
   b. les fichiers de configuration de l'AC ;
   c. le certificat de l'application généré par l'AC ;
   d. les sources de vos différents programmes Python ;
   e. un exemple d'attestation réalisée avec votre application qui puisse être vérifié par votre programme « `ExtrairePreuve` » ;
   f. le rapport ;