

FACULTÉ DES SCIENCES ET TECHNIQUES  
DE LIMOGES



MASTER 1 INFORMATIQUE

---

# Projet Sécurité des Usages des TIC

---

*Réalisé par:*

Amadou Oury DIALLO

Kilani WAJDI

# 1 Présentation des fonctionnalités

## 1.1 Création et Configuration de l'AC

L'AC est l'autorité de certification qui permet de certifier les certificats délivrés à un tiers qui est CertiPlus dans notre cas et qui est aussi propriétaire de l'AC. Nous avons procédé comme dans le TP en utilisant des fichiers de configurations ci-dessous :

- fichier de configuration de l'AC *root-ca.cnf*

```
[ req ]
default_bits = 256
default_keyfile = ecc.root.key.pem
distinguished_name = req_distinguished_name
x509_extensions = v3_ca
string_mask = nombstr
req_extensions = v3_req
[ req_distinguished_name ]
countryName = Country Name (2 letter code)
countryName_default = FR
countryName_min = 2
countryName_max = 2
stateOrProvinceName = State or Province Name (full name)
stateOrProvinceName_default = FRANCE
localityName = Locality Name (eg, city)
localityName_default = Limoges
0.organizationName = Organization Name (eg, company)
0.organizationName_default = CertifPlus
organizationalUnitName = Organizational Unit Name (eg,
    section)
organizationalUnitName_default = Service de Certification
commonName = Common Name (eg, Mon Autorite de Certification)
commonName_default = CertifPlus AC
commonName_max = 64
emailAddress = Email Address (eg, celle du responsable)
emailAddress_max = 40
[ v3_ca ]
basicConstraints = critical,CA:true
subjectKeyIdentifier = hash
[ v3_req ]
nsCertType = email,server
```

Ici nous avons choisi une de taille de clé par défaut de 256 bits qui est une recommandation pour des clés de types courbes elliptiques et

il utilise une clé : ***defaultkeyfile = ecc.root.key.pem***

- **Creation de la Bi-clé de l'AC : *ecc.root.key.pem***

```
openssl ecparam -out ecc.root.key.pem -name prime256v1 -  
genkey
```

- **Auto signature de l'AC :**

```
openssl req -new -x509 -days 7300 -config root-ca.cnf -key  
ecc.root.key.pem -out certiplus.root.crt
```

Maintenant nous disposons d'un certificat root (AC) : ***certiplus.root.crt*** et d'une clé privée : ***ecc.root.key.pem*** permettant respectivement de verifier et de signer d'autres certificats.

- **fichier de configuration pour l'obtention d'un certificat via l'AC : *appli.key.pem***

```
[ req ]  
default_bits = 256  
default_keyfile = appli.key.pem  
distinguished_name = req_distinguished_name  
string_mask = nombstr  
req_extensions = v3_req  
[ req_distinguished_name ]  
commonName = Common Name (eg, Jean DUPONT)  
commonName_max = 64  
emailAddress = Email Address (eg, prenom.nom@etu.unilim.fr)  
emailAddress_max = 40  
[ v3_req ]  
nsCertType = client, email  
basicConstraints = critical,CA:false
```

Nous avons choisi aussi une taille de clé de 256 bits de types courbe elliptique généré avec la commande openssl ci-dessous.

- **Génération de la Bi-clé de l'application de CertiPlus et de la demande de certification : *appli.key.pem* et *appli.crt***

```
openssl ecparam -out appli.key.pem -name prime256v1 -genkey  
openssl req -new -config appli.cnf -key appli.key.pem -out  
appli.crt
```

- Fichier de configuration pour la demande de certification par l'AC : *appli-sign.cnf*

```
[ ca ]
default_ca = default_CA
[ default_CA ]
dir = .
certs = $dir
new_certs_dir = $dir/newcerts
database = index.txt
serial = serial.txt
RANDFILE = seed.rnd
certificate = certiplus.root.crt
private_key = ecc.root.key.pem
default_days = 3650
default_crl_days = 30
default_md = sha256
preserve = yes
x509_extensions = appli
policy = policy_anything
[ policy_anything ]
commonName = supplied
emailAddress = supplied
[ appli ]
subjectAltName = email:copy
basicConstraints = critical ,CA: false
authorityKeyIdentifier = keyid:always
subjectKeyIdentifier = hash
extendedKeyUsage = clientAuth ,emailProtection
```

Utilisation du certificat ***certiplus.root.crt*** et de la clé privée ***ecc.root.key.pem*** pour la réalisation de la demande qui sera utilisé avec la commande openssl suivante pour obtenir un certificat signé et vérifié par l'AC :

```
openssl ca -config appli-sign.cnf -out certiplus.crt -batch
-infiles appli.crt
```

A partir de là, nous avons un certificat pour notre application certiplus valide signé et vérifié par l'AC.

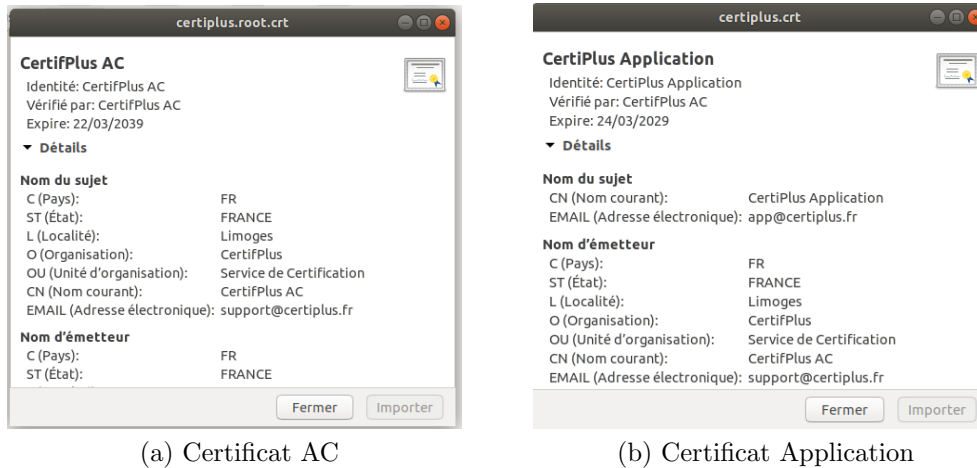


Figure 1

## 1.2 Le programme CreerOTP.py

Ce programme écrit en python3 ici est basé sur l'algorithme de google OTP, et permet d'obtenir un token valide pendant 30 s pour s'assurer qu'un utilisateur connecté sur l'application cliente est bien légitime en vérifiant le secret partagé entre les deux applications. Cet programme CreerOTP demande un secret à l'utilisateur puis à partir du timestamp auquel il est exécuté calcule un token OTP qui sera vérifié par le programme CreerAttestation.py ci-dessous pour s'assurer que l'utilisateur connaît bien le secret partagé. Les codes sources de ce programme se trouvent dans le fichier : **OTP.py** et pour obtenir un token OTP il faut lancer le programme **CreerOTP.py**

## 1.3 Le programme CreerAttestation.py

Ce programme récupère un token OTP en arguments calcule à son tour un token OTP à partir de son secret à lui, ensuite il compare son token à celui qu'il a reçu en argument, si les 2 correspondent alors l'utilisateur est légitime et sera autorisé à créer une attestation en fournissant les informations nécessaires sinon l'utilisateur est rejeté.

Ce programme une fois avoir autorisé l'utilisateur à fournir les informations pour la création d'une attestation sécurisée effectue :

- Création de l'image en exécutant les fonctions : **IntegrerTexte** et

**IntegrerQr** qui permettent respectivement d'intégrer un texte et un QrCode dans une image. Le QrCode contient des caractères obtenu à partir de la transformation de la signature des informations fournis au programme en base64. Pour effectuer cette signature le programme exécute la commande openssl suivante en servant se de la clé privée de l'application pour réaliser cette dernière :

```
openssl dgst -sha256 -sign appli.key.pem -out signature.sig  
fiche.txt
```

le fichier **signature.sig** obtenu après l'exécution de la commande ci-dessus est transformé en base64 pour être intégré dans le QrCode qui sera intégré dans l'image.

- Obtention d'une estampille à partir d'un serveur d'horodatage avec la fonction **Horodatage** qui prend une image contenant les informations intégré puis à partir d'un serveur d'horodatage obtient un timestamp de création de cette image. Les principales commandes exécutées sont :

#### Création de la requête pour le serveur d'horodatage

```
openssl ts -query -data attestation.png -no_nonce -sha512 -  
out requete.tsq
```

#### Obtention du timestamp

```
curl -H "Content-Type: application/timestamp-query" --data-  
binary '@requete.tsq' https://freetsa.org/tsr >  
timestamp.tsr
```

#### Vérification du Timestamp

```
openssl ts -verify -data attestation.png -in timestamp.tsr  
-CAfile cacert.pem -untrusted tsa.crt
```

- Intégration du bloc d'informations et Timestamp par steganographie : Il construit le bloc d'informations composé de la concaténation des informations personnelles fournies complétées par des ":" pour obtenir 64

octets et du bloc de timestamp obtenu par le serveur d'horodatage encodé en base64 ensuite ce bloc est caché par sténographie dans l'image.

- Envoi de mail sécurisé : le programme se connecte à un serveur de mail smtp dans notre cas le serveur de l'université smtp.unilim.fr ensuite il signe le mail avant de procéder l'envoi par la commande ci-dessous :

```
openssl cms -sign -in mail.txt -signer certiplus.crt -inkey  
  appli.key.pem -from amadou-oury.diallo@etu.unilim.fr -to  
  wajdi.kilani@etu.unilim.fr -subject "Certiplus -  
  Diplome signer" -out mail.msg
```

## 1.4 Le programme ExtrairePreuve.py

Ce programme prend en argument une image et à partir des fonctionnalités implémentés il effectue :

- Extraction du texte caché par stéganographie et reconstruit le bloc d'informations personnelles et converti le bloc timestamp encodé en base64 en sa forme initiale.
- Extraction et vérification du Qrcode : à Partir du bloc d'information extraite le programme vérifie la signature de celui-ci en extrayant la signature contenu dans le qrcode pour effectué une comparaison à partir des commandes ci-dessous. **Extraction de la clé publique du certificat de certiplus**

```
openssl x509 -pubkey -noout -in certiplus.crt > appli.  
  publickey.pem
```

### Vérification de la signature

```
openssl dgst -sha256 -verify appli.publickey.pem -signature  
  signature.out blocInfos.txt
```

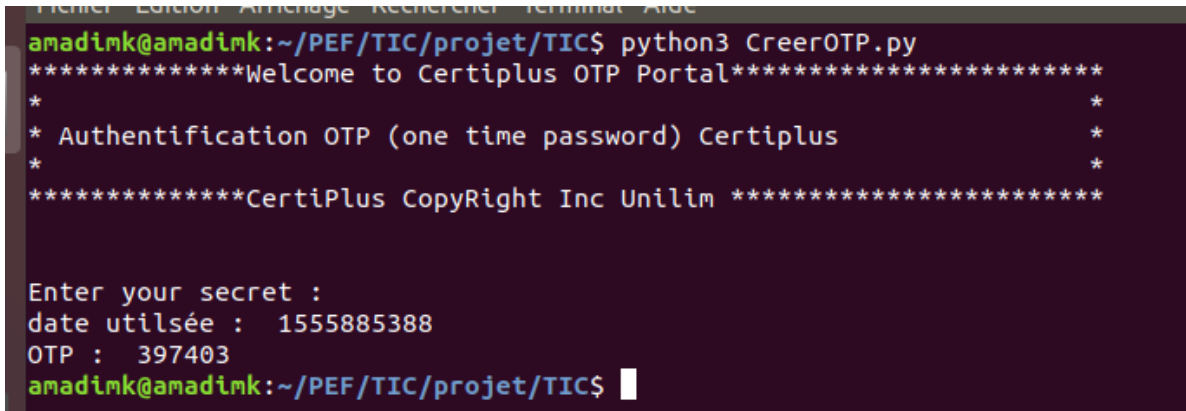
Ensuite, à partir du fichier timestamp reconstruit il procède à une vérification au niveau du serveur d'horodatage :

```
openssl ts -verify -in blocTimestamp.tsr -queryfile requete
.tsq -CAfile cacert.pem -untrusted tsa.crt
```

Enfin, si les données contenus dans l'image passe les deux vérifications alors l'image est valide.

## 1.5 Usages des programmes

- Générer un token otp : **python3 CreerOTP.py**  
Le secret : **CERTIPLUS**



```
amadi@amadi:~/PEF/TIC/projet/TIC$ python3 CreerOTP.py
*****Welcome to Certiplus OTP Portal*****
*
* Authentication OTP (one time password) Certiplus
*
*****CertiPlus CopyRight Inc Unilim *****

Enter your secret :
date utilisée : 1555885388
OTP : 397403
amadi@amadi:~/PEF/TIC/projet/TIC$
```

- Créer attestation : **python3 CreerAttestation.py 397403**



```

amadimk@amadimk:~/PEF/TIC/projet/TIC$ python3 CreerAttestation.py 397403
*****Welcome to CertiPlus Portal*****
*
* application destiné à la certification des diplômes certiplus *
*
*****CertiPlus CopyRight Inc Unilim *****
Nom : Diallo
Prenom : Amadou oury
Intitulé [Certification delivré à]: Diplome de cryptis
Mail : amadimk@gmail.com
Signature ...Done : AC/signed.DIALLO.sig
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 24748 100 24621 100 127 32567 167 --:--:-- --:--:-- --:--:-- 32692
Integration texte....Done : texte.png
Generation du Qrcode... done : qrcode.png
Combinaison des fichiers : texte.png et fond_attestation.png
Combinaison des fichiers : qrcode.png et combinaison.png
Using configuration from /usr/lib/ssl/openssl.cnf
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 1421 0 1333 100 88 901 59 0:00:01 0:00:01 --:--:-- 960
Using configuration from /usr/lib/ssl/openssl.cnf
Horodatage effectué
Traitement de l'image ....
Insertion des données complétés...!
Envoi du mail en cours ...!!!
Votre identifiant Unilim pour l'envoi du mail : diallo87
votre mot de passe :
Mail envoyé...!!!

```

- Extraire Preuve : `python3 ExtrairePreuve.py final-attestation.png`

```

Fichier Edition Affichage Recherche Terminal Aide
amadimk@amadimk:~/PEF/TIC/projet/TIC$ python3 ExtrairePreuve.py final_attestation.png
Extraction de la signature dans le Qrcode ...!!!
Lancement de la procedure de verification de la signature ...
La verification de la signature à reussie avec succès !!!
Lancement de la verification de l'horodatage ...!!!
Using configuration from /usr/lib/ssl/openssl.cnf
La verification de l'horodatage à reussie avec succès !!!
L'image est verifié et est valide
amadimk@amadimk:~/PEF/TIC/projet/TIC$

```

## 1.6 Analyses des risques

En analysant notre procédé nous avons déterminer plusieurs actifs pouvant créer des risques de dysfonctionnement ou des usages frauduleuses s'ils ne

sont pas protégés :

**Actifs primaires :**

- les clés privées ( AC et application )
- le secret partagé
- le service smtp
- le service d'horodatage
- les certificats

**Actifs secondaires :**

- les fichiers de configuration de l'AC
- les fichiers contenant les sources du programme CreerAttestation
- les fichiers de requête du timestamp

Les différentes menaces pouvant survenir sur ces actifs sont :

- perte, vol et usage frauduleuse des clés privées
- reverse engineering sur le code source du programme CreerAttestation qui contient le secret partagé
- Indisponibilité de la personne connaissant le secret partagé (décès, mort etc.) qui est le seul à le connaître.
- Indisponibilité des serveurs smtp et/ou horodatage
- perte du fichier de la requête du timestamp utile pour la vérification du timestamp au près du serveur d'horodatage.

**Relations actifs primaires et secondaires :**

- les clés privées ainsi que les certificats sont inclus dans les fichiers de configurations de l'AC
- le secret partagé est contenu en clair dans le fichier contenant le code source du programme CreerAttestation