

ISEN

ALL IS DIGITAL!



yncréa

PROJET CRYPTOGRAPHIE



SOMMAIRE

1	Introduction.....	3
1.1	Présentation générale	3
1.2	Arborescence du projet	3
2	Création du certificat.....	4
2.1	Création du certificat auto-signé	4
2.2	Ligne de commande pour la CA.....	5
2.3	Création du certificat intermédiaire	6
2.4	Base OSCP.....	7
3	Page web	7
3.1	Introduction.....	7
3.2	Composition du site.....	7
3.3	Page menu	8
3.4	Page Création de certificat	8
3.5	Page suppression de certificat.....	10
3.6	Page base de données	10
3.7	Page base de données ocsp.....	11
4	Base De Données MySQL	11
4.1	Base de données MySQL	11
4.2	Création de la base de données et du user	11
5	Gestion des mails	12
5.1	Gestion des mails.....	12
6	Test de nos certificats.....	14
6.1	Signature réussie	14
6.2	Echec de la signature	15
7	Téléchargement du certificat	16
7.1	Contenu	16
8	Guide d'utilisation	17
8.1	Guide	17
9	Conclusion	19
10	Sources	19

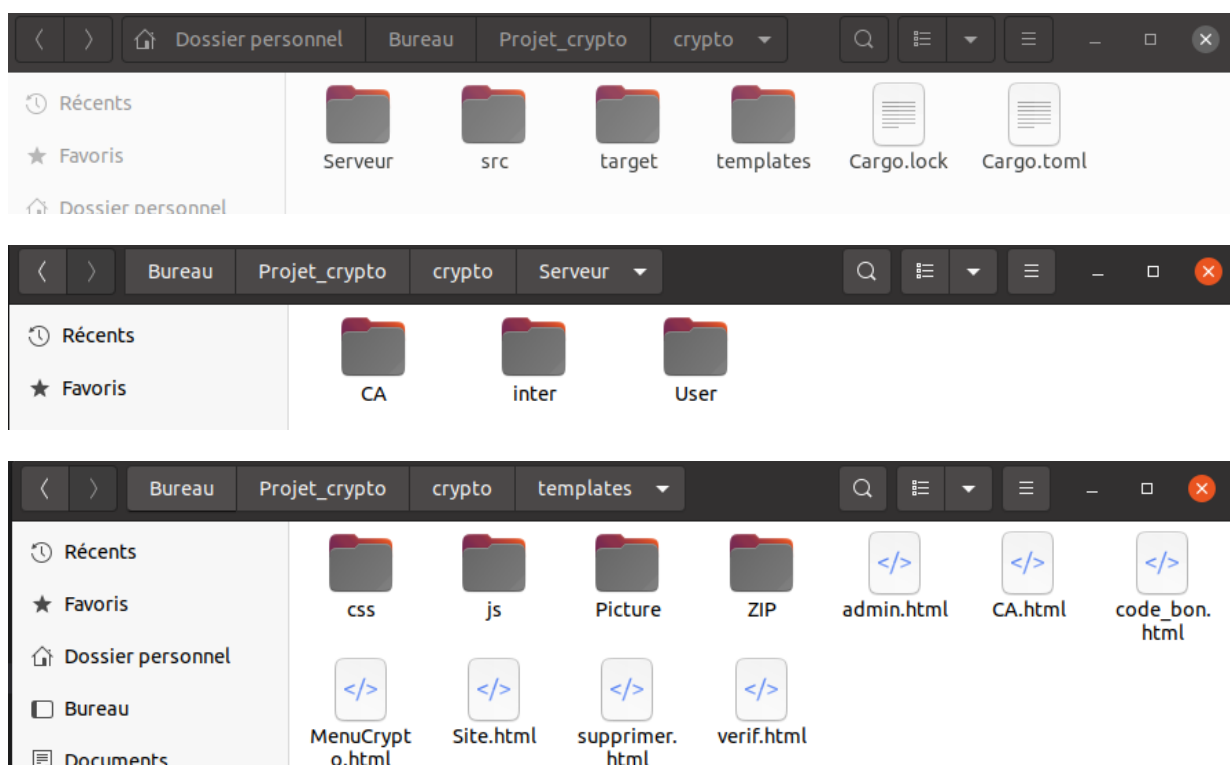
1 INTRODUCTION

1.1 PRESENTATION GENERALE

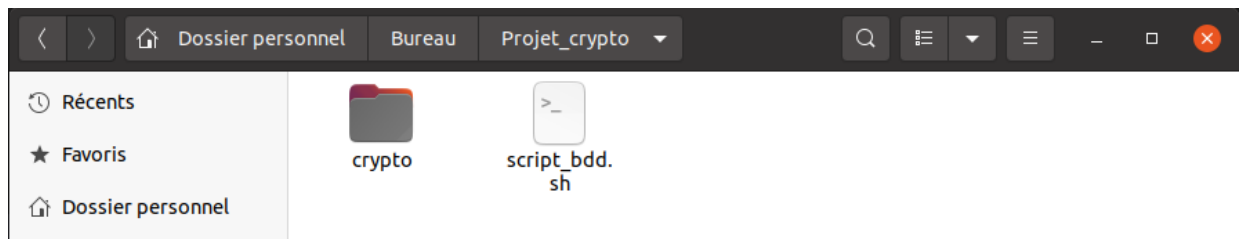
L'objectif de ce projet est de créer une autorité de certification permettant de générer des certificats pour des utilisateurs désirant signer leurs mails. Pour réaliser ce projet nous avons utilisé OpenSSL qui va permettre de créer le certificat ainsi que l'utilisation du langage Rust. Nous avons utilisé le langage Rust car c'est un langage sécurisé et il offre beaucoup de librairie notamment avec actix-web pour l'interface utilisateur. Nous avons également utilisé MySQL pour visualiser les utilisateurs aillant un certificat.

1.2 ARBORESCENCE DU PROJET

Concernant l'arborescence, le projet est composé de plusieurs fichiers. Nous avons le dossier serveur avec tous les fichiers concernant OpenSSL, Le fichier src avec le code main.rs, le fichier target utilisé par Rust pour stocker les fichiers de sortie générés lors de la compilation du code source en un exécutable ou une bibliothèque. Il est créé lors de la première compilation du code c'est pour cela qu'il n'est pas présent dans le zip. Le dossier templates composé de toutes les photos, fichiers css, js et html qui constitue le site web. Et enfin les dossiers Cargo.lock et Cargo.toml pour gérer toutes les dépendances utilisées dans le main.rs.



Nous avons également un script Bash qui va permettre de créer la base de données. Ce code sera à exécuter avant le projet.



2 CREATION DU CERTIFICAT

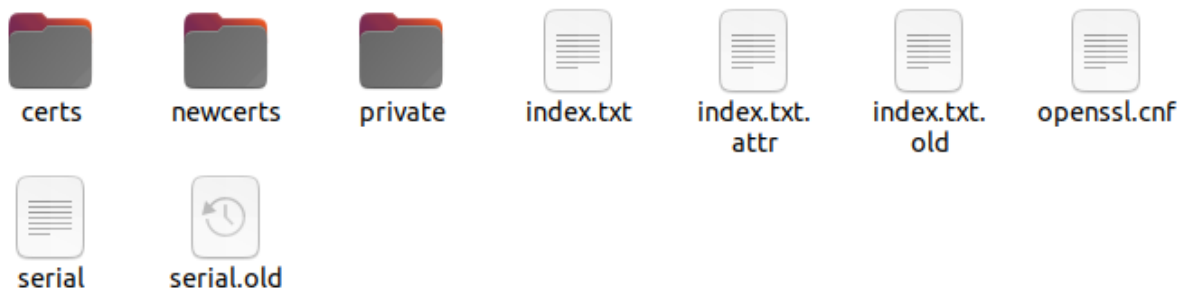
2.1 CREATION DU CERTIFICAT AUTO-SIGNE

Pour créer nos certificats nous avons utilisé OpenSSL un outil permettant de créer des certificats. En Rust nous pouvons insérer des lignes de commande dans le code ce qui nous a facilité la tâche.

```
let _output = Command::new("openssl")
    .arg("genpkey")
    .arg("-algorithm")
    .arg("EC")
    .arg("-pkeyopt")
    .arg("ec_paramgen_curve:prime256v1")
    .arg("-out")
    .arg(format!("Serveur/User/{}/{}.key.pem", nom, nom))
    .output()
    .expect("La commande a échoué");
```

Avant la création du certificat, nous avons mis en place un répertoire visant à contenir le nécessaire :

- un répertoire certs, qui contiendra le certificat de l'ACR
- un répertoire crt, contenant la CRL (la liste des certificats révoqués)
- un répertoire newcerts, qui contiendra les certificats signés par l'ACR.
- un répertoire private, qui va contenir la clef privée de l'ACR
- un fichier index.txt, qui va servir de base de données
- un fichier openssl.conf, le fichier
- un fichier serial, qui contient le prochain numéro de série à utiliser pour la prochaine signature de certificat (1000 de base)



2.2 LIGNE DE COMMANDE POUR LA CA

N'ayant pas de clé USB a vous envoyé nous avons réalisé ces étapes directement dans nos répertoires sur le projet.

1. Première commande : Création de la clé privée :

```
openssl genpkey -algorithm EC -pkeyopt ec_paramgen_curve:prime256v1 -out ca.key.pem
```

Cette commande permet de créer une clef privée en utilisant une courbe elliptique (prime256v1). Nous avons utilisé une courbe elliptique car c'est plus robuste que sha256.

2. Création du fichier de configuration openssl.cnf :

[CA_default] : Cette partie définit plusieurs paramètres de base

```
[ CA_default ]
dir               = /home/drets/Documents/ProjetCrypto3/ca
certs             = $dir/certs
crl_dir           = $dir/crl
new_certs_dir     = $dir/newcerts
database         = $dir/index.txt
serial           = $dir/serial
RANDFILE         = $dir/private/.rand

private_key       = $dir/private/ca.key.pem
certificate       = $dir/certs/ca.cert.pem

crlnumber        = $dir/crlnumber
crl               = $dir/crl/ca.crl.pem
crl_extensions   = crl_ext
default_crl_days = 30

default_md       = sha256

name_opt         = ca_default
cert_opt        = ca_default
preserve        = no
policy          = policy_strict
```

-dir correspond aux répertoires utilisés par la CA

-policy correspond à la politique à utiliser, différente pour la CA et la CI (voir ci-dessous)

[policy_strict] : Informations à fournir pour la certification

```
[ policy_strict ]
stateOrProvinceName = optional
localityName        = optional
organizationName    = optional
organizationalUnitName = optional
commonName          = supplied
emailAddress         = optional
```

-supplied signifie que l'utilisateur doit obligatoirement fournir l'information

[v3_ca] : Extension pour la CA

```
[ v3_intermediate_ca ]
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = critical, CA:true, pathlen:0
keyUsage = critical, digitalSignature, cRLSign, keyCertSign
```

-CA :true signifie que le certificat est une CA

-keyUsage définit tous les usages possibles, notamment ici la révocation et la signature

3. Autosignature du certificat :

```
loica@loica-VirtualBox:~$ openssl req -config openssl.cnf -key private/ca.key.pem -new -x509 -extensions v3_ca -out certs/ca.cert.pem -subj "/C=pays/ST=région/L=ville/O=organisation/OU=service/CN=nom/emailAddress=mail"
```

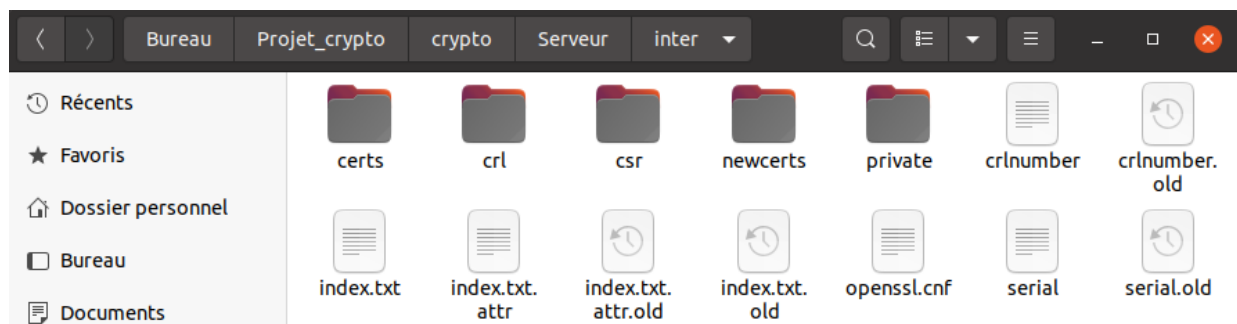
Le pays, région, ville, organisation, service, nom, adresse est fourni par l'utilisateur via la page web. Cependant il est important de rentrer le pays en deux lettres exemple 'FR'.

Après toutes ces étapes votre CA est créée et doit être conservé dans un fichier du projet.

2.3 CREATION DU CERTIFICAT INTERMEDIAIRE

Comme pour la CA nous avons créé plusieurs fichiers :

- Création d'un répertoire pour le certificat intermédiaire, sur le modèle ci-dessus (avec un répertoire csr contenant la requête de signature à la CA). De plus le fichier de configuration est légèrement différent au niveau de la policy.



-Création de la clef (même requête que pour la CA)

-Création de la requête :

```
loica@loica-VirtualBox:~$ openssl req -config openssl.cnf -new -key private/inter.key.pem -out csr/inter.csr.pem -subj "/C=pays2/ST=région2/L=ville2/O=organisation2/OU=service2/CN=nom2/emailAddress=mail2"
```

-Signature du certificat par la CA :

```
loica@loica-VirtualBox:~$ openssl ca -config ../ca/openssl.cnf -extensions v3_intermediate_ca -days 3650 -notext -in csr/inter.csr.pem -out certs/inter.cert.pem
```

Nous avons maintenant notre CA et CI, il faut maintenant les utiliser pour créer des certificats pour chacun des users qui utilise notre site.

2.4 BASE OSCP

La base contient tous les certificats signés, et notamment leur numéro de série, leur état, valide ou révoqué (V ou R) et les informations entrées par l'utilisateur. Le fichier index.txt contient le numéro de série du prochain certificat signé.

Pour faire la révocation de certificat nous avons utilisé cette commande :

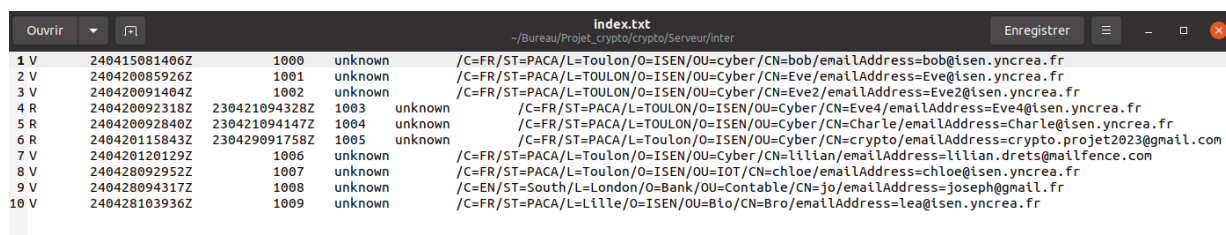
-Génération de la crl :

```
openssl ca -config inter/openssl.cnf -gencrl -out inter/crl/inter.crl.pem
```

Concernant la révocation de certificat nous avons utilisé la commande suivante :

```
openssl ca -config inter/openssl.cnf -revoke user/nom/nom.cert.pem
```

Ci-dessous voici un exemple de notre fichier index.txt avec les certificats révoqués et les certificats valides depuis la création de notre site.



État	Série	Numéro	Statut	Informations
1 V	240415081406Z	1000	unknown	/C=FR/ST=PACA/L=Toulon/O=ISEN/OU=cyber/CN=bob/emailAddress=bob@isen.yncrea.fr
2 V	240420085926Z	1001	unknown	/C=FR/ST=PACA/L=Toulon/O=ISEN/OU=Cyber/CN=Eve/emailAddress=Eve@isen.yncrea.fr
3 V	240420091404Z	1002	unknown	/C=FR/ST=PACA/L=Toulon/O=ISEN/OU=Cyber/CN=Eve2/emailAddress=Eve2@isen.yncrea.fr
4 R	240420092318Z	230421094328Z	1003 unknown	/C=FR/ST=PACA/L=Toulon/O=ISEN/OU=Cyber/CN=Eve4/emailAddress=Eve4@isen.yncrea.fr
5 R	240420092840Z	230421094147Z	1004 unknown	/C=FR/ST=PACA/L=Toulon/O=ISEN/OU=Cyber/CN=Charle/emailAddress=Charle@isen.yncrea.fr
6 R	240420115843Z	230429091758Z	1005 unknown	/C=FR/ST=PACA/L=Toulon/O=ISEN/OU=Cyber/CN=crypto/emailAddress=crypto.projet2023@gmail.com
7 V	240420120129Z	1006	unknown	/C=FR/ST=PACA/L=Toulon/O=ISEN/OU=Cyber/CN=lilian/emailAddress=lilian.drets@mailfence.com
8 V	240420092952Z	1007	unknown	/C=FR/ST=PACA/L=Toulon/O=ISEN/OU=IoT/CN=chloe/emailAddress=chloe@isen.yncrea.fr
9 V	240420094317Z	1008	unknown	/C=EN/ST=South/L=London/O=Bank/OU=Contable/CN=jo/emailAddress=joseph@gmail.fr
10 V	240420103936Z	1009	unknown	/C=FR/ST=PACA/L=Lille/O=ISEN/OU=Bio/CN=Bro/emailAddress=lea@isen.yncrea.fr

3 PAGE WEB

3.1 INTRODUCTION

Pour réaliser le frontend, nous sommes partis sur la réalisation d'un site web en html css et javascript qui est hébergé sur actix web un Framework Rust qui permet d'ouvrir un localhost sur un port souhaité. Dans notre cas, y mettre notre site web tout en aillant une connexion avec le backend en Rust.

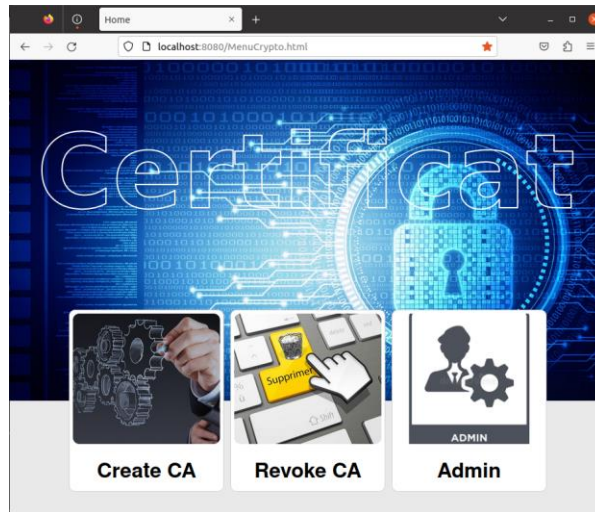
Le port utilisé pour notre projet est le port 8080 à l'adresse suivant : <http://localhost:8080/>. Dans le cas d'une utilisation réelle le projet sera hébergé sur un véritable serveur.

3.2 COMPOSITION DU SITE

Notre site web est composé d'une page menu, d'une page pour supprimer les certificats, une page pour voir la base de données accessible uniquement par l'administrateur, une page admin, une page permettant la demande de certificat ainsi que la vérification de l'adresse électronique.

3.3 PAGE MENU

Cette page est la page principale du projet elle permet d'accéder à la page suppression de certificat, la page admin ainsi que la page création de certificat.



3.4 PAGE CREATION DE CERTIFICAT

Pour créer un certificat l'utilisateur arrive tout d'abord sur une page où il doit renseigner son adresse mail afin de faire une vérification de l'adresse mail et enfin une page demande de certificat où l'utilisateur va pouvoir renseigner ses informations.

Renseignement d'adresse mail

Adresse mail:

Soumettre

© 2023 Projet Cryptographie Lilian Drets Loïc Allegretti

Sur le backend il y a une fonction qui récupère l'adresse électronique de l'utilisateur pour lui envoyer un code aléatoire

← → ↻ localhost:8080/verif.html ☆ 🛡️ 📄 ☰

Vérification de l'adresse mail

Code de vérification:

Vérification

© 2023 Projet Cryptographie Lilian Drets Loïc Allegretti

Ici l'utilisateur doit rentrer le code reçu par mail s'il est faux il sera redirigé vers la page menu sinon vers la page de la création de certificat.

← → ↻ 127.0.0.1:8080/CA.html ☆ 🛡️ 📄 ☰

Création du Certificat

Nom :

Prénom :

Pays :

Région :

Ville :

Organisation :

Service :

Adresse mail :

Mot de passe pour votre certificat :

Soumettre

© 2023 Projet Cryptographie Lilian Drets Loïc Allegretti


3.5 PAGE SUPPRESSION DE CERTIFICAT

Sur la page suppression tout se fait en backend l'utilisateur a juste à rentrer les informations demandées pour supprimer le certificat sur la base de données et sur la base OCSP.



3.6 PAGE BASE DE DONNEES

Cette page permet de visualiser l'ensemble des utilisateurs aillant demander un certificat sur le site.



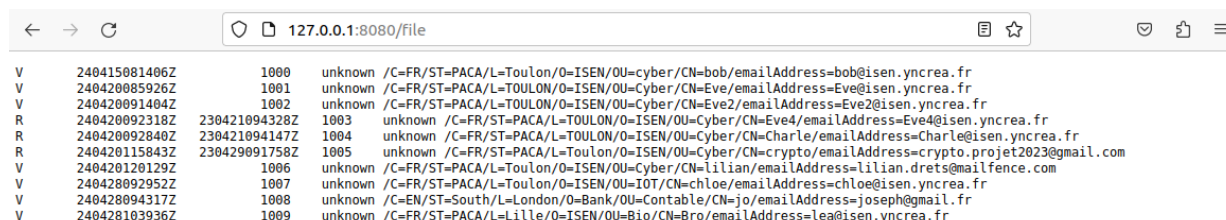
Nom	Prenom	Code	Adresse	pays	ville	organisation
bob	bob	15489	bob@isen.yncrea.fr	FR	Toulon	ISEN
Eve	Eve	24589	Eve@isen.yncrea.fr	FR	Toulon	ISEN
Eve2	Eve2	89562	Eve2@isen.yncrea.fr	FR	Toulon	ISEN
drets	lilian	64521	lilian@mailfence.com	FR	Toulon	ISEN
chloe	chloe	25828	chloe@isen.yncrea.fr	FR	Toulon	ISEN
jo	joseph	24667	joseph@gmail.com	EN	London	Bank
bro	lea	17757	lea@isen.yncrea.fr	FR	Lille	ISEN

Pour accéder à cette base de données nous avons fait le choix de mettre une page admin avec un code et un mot de passe pour éviter que les utilisateurs voient les certificats des autres. Le username est CRYPTO et le mot de passe CRYPTO.



3.7 PAGE BASE DE DONNEES OCSP

Nous avons créé une page qui ouvre le fichier index.txt qui est le fichier rassemblant tous les certificats créés et supprimés depuis notre site web. L'avantage comparé à la base de données MySQL c'est que nous pouvons voir les certificats supprimés.



V	240415081406Z	1000	unknown	/C=FR/ST=PACA/L=Toulon/O=ISEN/OU=cyber/CN=bob/emailAddress=bob@isen.yncrea.fr
V	240420085926Z	1001	unknown	/C=FR/ST=PACA/L=TOULON/O=ISEN/OU=Cyber/CN=Eve/emailAddress=Eve@isen.yncrea.fr
V	240420091404Z	1002	unknown	/C=FR/ST=PACA/L=TOULON/O=ISEN/OU=Cyber/CN=Eve2/emailAddress=Eve2@isen.yncrea.fr
R	240420092318Z	230421094328Z	1003	unknown /C=FR/ST=PACA/L=TOULON/O=ISEN/OU=Cyber/CN=Eve4/emailAddress=Eve4@isen.yncrea.fr
R	240420092840Z	230421094147Z	1004	unknown /C=FR/ST=PACA/L=TOULON/O=ISEN/OU=Cyber/CN=Charle/emailAddress=Charle@isen.yncrea.fr
R	240420115843Z	230429091758Z	1005	unknown /C=FR/ST=PACA/L=Toulon/O=ISEN/OU=Cyber/CN=crypto/emailAddress=crypto.projet2023@gmail.com
V	240420120129Z	1006	unknown	/C=FR/ST=PACA/L=Toulon/O=ISEN/OU=Cyber/CN=lilian/emailAddress=lilian.drets@mailfence.com
V	240420092952Z	1007	unknown	/C=FR/ST=PACA/L=Toulon/O=ISEN/OU=IoT/CN=chloe/emailAddress=chloe@isen.yncrea.fr
V	240420094317Z	1008	unknown	/C=EN/ST=South/L=London/O=Bank/OU=Contable/CN=jo/emailAddress=joseph@gmail.fr
V	240420103936Z	1009	unknown	/C=FR/ST=PACA/L=Lille/O=ISEN/OU=Bio/CN=Bro/emailAddress=lea@isen.yncrea.fr

4 BASE DE DONNEES MYSQL

4.1 BASE DE DONNEES MYSQL

Pour gérer les utilisateurs sur notre site web nous avons décidé de créer une base de données MySQL afin de pouvoir visualiser nos ajouts et suppressions de certificats. Pour cela nous avons créé une base de données utilisateurs avec l'utilisateur crypto ayant tous les droits afin que nous puissions à l'aide de ligne de commande dans le code ajouter et supprimer des utilisateurs. Dans cette base de données le nom, prénom, code, adresse mail, pays, ville et l'organisation de la personne créant le certificat y seront renseignés.

Exemple ci-dessous de comment supprimer des utilisateurs à l'aide d'une commande MySQL en Rust :

```
let mut conn = db.get_conn().unwrap();
conn.exec_drop(
    r"DELETE FROM utilisateurs WHERE nom = :nom AND prenom = :prenom AND code = :code",
    params! {
        "nom" => nom,
        "prenom" => prenom,
        "code" => code,
    },
).unwrap();
```

4.2 CREATION DE LA BASE DE DONNEES ET DU USER

Avant de lancer le code il est impératif de créer la base de données afin de pouvoir utiliser le site sans erreur. Pour cela nous avons eu l'idée de créer un petit script Bash à exécuter avant de lancer le code sur votre terminale. Ce script va vous créer la table crypto, la base de données utilisateurs ainsi qu'un utilisateur crypto ayant tous les droits sur cette base de données.

```
#!/bin/bash

# Vérifier si MySQL est installé
if ! command -v mysql &> /dev/null
then
    echo "MySQL n'est pas installé, en cours d'installation ..."
    sudo apt install mysql-server
fi

# Connexion à MySQL et exécution des requêtes
mysql -u root -p -e "
CREATE DATABASE IF NOT EXISTS crypto;
USE crypto;
CREATE TABLE IF NOT EXISTS utilisateurs (nom VARCHAR(50), prenom VARCHAR(50), code VARCHAR(50), adresse VARCHAR(50), pays VARCHAR(50), ville
VARCHAR(50), organisation VARCHAR(50));
CREATE USER 'crypto'@'localhost' IDENTIFIED BY 'crypto';
GRANT ALL PRIVILEGES ON crypto.* TO 'crypto'@'localhost';
"
```

Ce petit script va exécuter toutes les commandes que nous avons réalisé pour créer cette base de données. Il va également vous installer MySQL si ce n'est pas déjà fait sur votre environnement.

5 GESTION DES MAILS

5.1 GESTION DES MAILS

Concernant les mails nous avons d'abord créer une adresse mailfence sur Thunderbird mais nous nous sommes rendu compte qu'au bout de 2 semaines il fallait payer pour la garder active. Nous avons donc créé une adresse mail Gmail pour le projet :

Dans notre code d'envoi de mail nous devons mettre le mot de passe en clair ainsi que le nom mais avec Gmail, une sécurité en plus est à effectuer pour pouvoir envoyer un mail. Cette sécurité est de créer un mot de passe qui autorise Rust à envoyer des mails avec l'adresse électronique crypto.

Pour cela nous somme aller dans :

Paramètres de l'adresse mail -> sécurité -> validation en deux étapes -> mot de passe d'application

Et nous avons créé un mot de passe de vérification pour notre adresse mail. Et comme vous pouvez le voir ci-dessous dans le code du mail, le nom d'utilisateur est marqué en clair mais le mot de passe est un mot de passe généré par Gmail ce qui nous évite de renseigner notre vrai mot de passe.


```
//Code pour envoyer le mail
let email = EmailBuilder::new()
.to(adresse)
.from("crypto.projet2023@gmail.com")
.subject("Example subject")
.text(number.to_string())
.build()
.unwrap();

let mut mailer = SmtplibClient::new_simple("smtp.gmail.com")
.unwrap()
.credentials(Credentials::new("crypto.projet2023".into(), "kwhu qdat yrsy zjqb".into()))
.transport();


let result = mailer.send(email.into());
println!("{:?}", result);
```







Voici un exemple de mail avec le code à renseigner sur le site :

Example subject




crypto.projet2023@gmail.com

À  Loïc ALLEGRETTI

sam. 10:19

Stratégie de rétention Junk Email (30 jours) Date d'expiration 29/05/2023

 Cet élément expirera dans 29 jours. Pour le conserver plus longtemps, appliquez une autre stratégie de rétention. Les liens et les autres fonctionnalités ont été désactivés dans ce message. Pour activer ces fonctionnalités, veuillez déplacer ce message dans la boîte de réception.

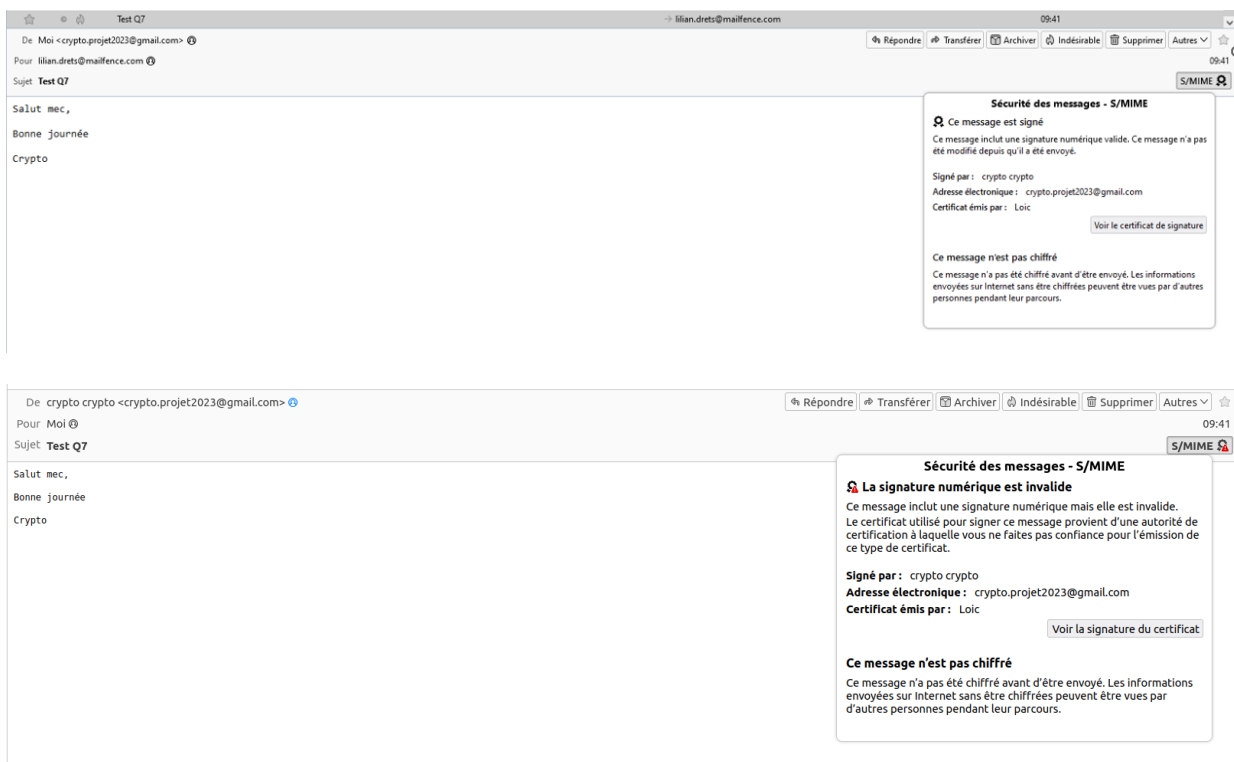
[Vous ne recevez pas souvent de courriers de crypto.projet2023@gmail.com. Découvrez pourquoi ceci est important à <https://aka.ms/LearnAboutSenderIdentification>]

29115



6.2 ECHEC DE LA SIGNATURE

Si le certificat n'est pas signé par une CA ou CI de confiance le mail sera envoyé mais ne sera pas signé correctement.



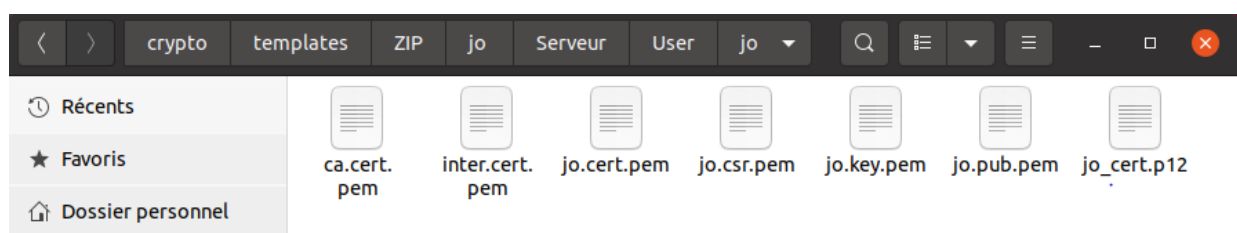


7 TELECHARGEMENT DU CERTIFICAT

7.1 CONTENU

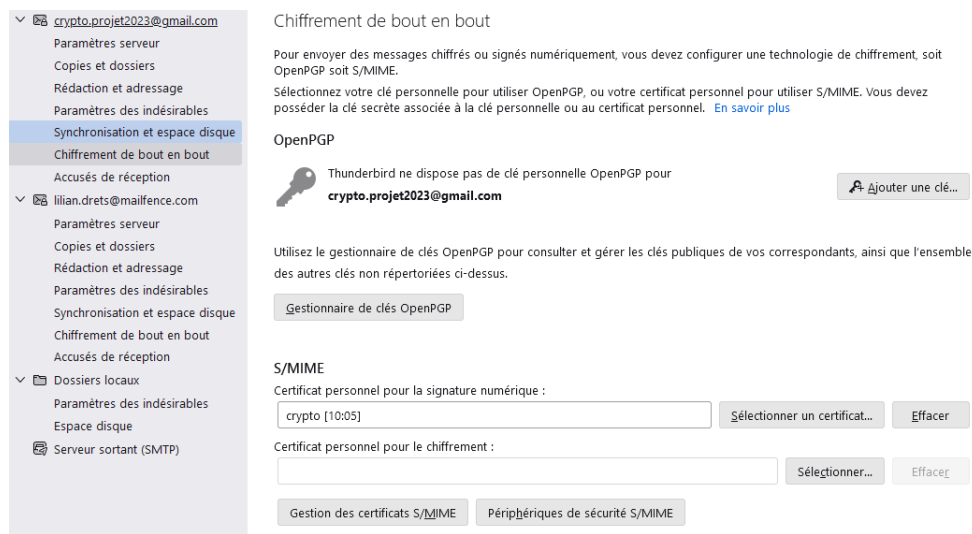
A la fin de la création de votre certificat un bouton téléchargement va vous permettre de télécharger votre propre certificat. Dans ce dossier il y a la CA la CI à mettre sur votre adresse mail ainsi que votre certificat vérifié par la CI à ajouter également sur votre boîte mail.

Exemple de fichier pour l'utilisateur nommé Jo :



Les informations importantes dans le dossier sont les fichiers ca.cert.pem, inter.cert.pem et jo_cert.p12 (thunderbird accepte que les .p12 comme certificat et non un .pem)

Après avoir récupéré votre certificat ajouter la CA et CI ainsi que votre certificat dans Thunderbird. Mettre la CA CI dans la case gestion des certificats et votre certificat dans sélectionner un certificat comme ci-dessous :



8 GUIDE D'UTILISATION

8.1 GUIDE

Avant de lancer le code assurez-vous d'avoir Rust sinon faites les commandes suivantes :

```
loic@loic-VirtualBox:~/Bureau/Projet_crypto/crypto$ snap install rustup --classic
rustup 1.24.3 par Daniel Silverstone (dsilvers) installé
loic@loic-VirtualBox:~/Bureau/Projet_crypto/crypto$ sudo apt install cargo
```

Assurez-vous également d'avoir OpenSSL sur votre environnement linux.

```
sudo apt-get install openssl
```

Ensuite lancez le code bash afin de créer la base de données. Faire `sudo ./script_bdd.sh`.

La dernière étape est de remplacer le chemin du dossier config.cnf par votre chemin sinon des problèmes interviendront lors de l'utilisation du projet. A la place de dir mettre le chemin où est situé votre dossier openssl.cnf. Si vous exécutez le projet sur votre bureau il faudra juste remplacer loica par votre nom (la fin n'est pas à modifier elle reste identique peu importe le lieu
'../Projet_crypto/crypto/Serveur/inter')

```
Ouvrir  ▾  [+]  openssl.cnf  ~/Bureau/Projet_crypto/crypto/Serveur/inter
1  ca
2  default_ca = CA_default
3
4  [ CA_default ]
5  dir                = /home/loica/Bureau/Projet_crypto/crypto/Serveur/inter
6  certs              = $dir/certs
7  crl_dir            = $dir/crl
8  new_certs_dir      = $dir/newcerts
9  database            = $dir/index.txt
10 serial             = $dir/serial
11 RANDFILE           = $dir/private/.rand
12
13 private_key         = $dir/private/inter.key.pem
14 certificate         = $dir/certs/inter.cert.pem
15
16 crlnumber           = $dir/crlnumber
17 crl                 = $dir/crl/ca.crl.pem
18 crl_extensions      = crl_ext
19 default_crl_days    = 30
20
21 default_md          = sha256
22
23 name_opt            = ca_default
24 cert_opt            = ca_default
25 preserve            = no
26 policy              = policy_loose
27
```

Lorsque toutes ces étapes sont réalisées, rendez vous dans le dossier crypto et faites un cargo build. Attendez l'installation des dépendances (cela peut prendre un petit moment) ensuite faites cargo run et rendez vous sur l'url suivante : <http://localhost:8080/MenuCrypto.html>. Vous êtes maintenant sur la page menu de notre projet vous pouvez ajouter supprimer et voir les certificats. Pour voir les certificats un mot de passe de vérification d'administrateur vous sera demandé. Il vous faut donc l'accès pour y aller ce qui évite que tout le monde puisse accéder aux certificats des autres (le code admin est CRYPTO CRYPTO).

Il est aussi intéressant d'avoir internet car certaines animations nécessitent internet pour être vu correctement.

9 CONCLUSION

Pour conclure grâce à ce projet nous avons pu manipuler ainsi que mieux comprendre la notion de certificat, l'amélioration de nos compétences en Rust, HTML et aussi une découverte d'OpenSSL. Concernant les améliorations dans le futur nous pourrions améliorer le code afin d'éviter les quelques pré-requis nécessaires pour lancer le code (pas d'installation, pas de fichier à modifier). Nous pouvons envisager d'améliorer le style du mail afin qu'il soit plus professionnel (pas un simple code mais une meilleure présentation) et également rendre notre projet utilisable sous Windows.

10 SOURCES

Pour réaliser ce projet nous nous sommes servis de YouTube pour mieux comprendre les CA, CI et OCSP. Nous avons également utilisé : <https://openssl-ca.readthedocs.io/en/latest/create-the-intermediate-pair.html> pour openssl et la documentation Rust.