



Université Abdelmalik Es-saadi
Faculté des Sciences et Techniques de Tanger
Département Mathématique

Mini projet

Cryptographie avec Openssl

*Spécialité : Master en Modélisation Mathématique et Sciences des
Données*

Projet

Mise en place d'une Infrastructure à Clés Publiques
(PKI) à Trois Niveaux

Encadré par

— Mme. LECCHHAB OUADRASSI Nihad

Réalisé par

— Sefiane El-kassimi

2024/2025

Table des matières

0.1	Introduction	2
0.2	Objectifs du projet	3
0.3	Description Fonctionnelle	3
0.4	Fonctionnalités attendues	4
0.5	Contraintes techniques et recommandations	4
0.6	Bibliothèques et outils utilisés	5
0.7	Interface Utilisateur Web (Flask)	6
0.7.1	Fonctionnalités principales de l'interface	6
0.7.2	Avantages de l'interface	7
0.7.3	Structure générale de l'interface	7
0.8	3.2 Architecture générale de l'application	9
0.8.1	Diagramme de classes (modèle métier)	9
0.8.2	Modèle de données	10
0.9	Implémentation du Projet	10
0.9.1	Génération de la clé privée et CSR	11
0.10	Captures d'écran de l'interface utilisateur	12
0.11	Conclusion	14

0.1 Introduction

Dans un contexte numérique en constante évolution, la protection des échanges d'informations est primordiale. L'Infrastructure à Clés Publiques (PKI) joue un rôle essentiel en garantissant la confidentialité, l'intégrité et l'authenticité des communications grâce à la gestion des certificats numériques. Ce projet consiste à déployer une PKI à trois niveaux (Root CA, Intermediate CA, certificats finaux) afin de maîtriser l'ensemble du cycle de vie des certificats : génération, signature, distribution, révocation. Une interface

web simplifie la gestion quotidienne de cette infrastructure complexe. Ce travail permet d’approfondir les notions de cryptographie appliquée et de sécurité numérique, tout en mettant en œuvre les standards internationaux comme X.509.

0.2 Objectifs du projet

Le principal objectif de ce projet est de concevoir et de mettre en œuvre une Infrastructure à Clés Publiques (PKI) à trois niveaux, tout en garantissant la sécurité, la fiabilité et la simplicité de gestion de l’ensemble du système. À travers cette réalisation, plusieurs objectifs spécifiques sont visés :

- Comprendre le fonctionnement et l’architecture d’une PKI hiérarchique.
- Générer et gérer des certificats numériques pour différents niveaux d’autorité.
- Implémenter la signature et la révocation de certificats à l’aide d’OpenSSL.
- Mettre en place une chaîne de confiance entre la Root CA, l’Intermediate CA et les certificats finaux.
- Assurer la sécurité des clés privées et le respect des bonnes pratiques cryptographiques.
- Créer une interface web conviviale pour faciliter la gestion de la PKI.
- Vérifier la validité des certificats et leur conformité aux normes X.509.
- Documenter toutes les étapes du projet, depuis la conception jusqu’à la démonstration.

0.3 Description Fonctionnelle

L’Infrastructure à Clés Publiques (PKI) mise en place dans ce projet repose sur une architecture hiérarchique à trois niveaux, permettant d’assurer une séparation des responsabilités et une meilleure sécurité. Cette hiérarchie se compose des éléments suivants :

- **Root CA (Autorité Racine)** : C’est l’autorité de confiance ultime. Elle est auto-signée et utilisée uniquement pour signer les certificats des autorités intermédiaires. Sa clé privée doit être fortement protégée et rarement utilisée afin de limiter les risques.
- **Intermediate CA (Autorité Intermédiaire)** : Elle est signée par la Root CA et est responsable de la signature des certificats finaux destinés aux utilisateurs,

serveurs ou autres entités. Cela permet de préserver l'intégrité de la Root CA.

- **Leaf Certificates (Certificats finaux)** : Ces certificats sont utilisés par des clients, des serveurs ou des applications. Ils permettent de prouver leur identité dans un environnement sécurisé (ex : HTTPS, VPN).

Cette organisation garantit une chaîne de confiance solide et modulaire. Les opérations principales de la PKI incluent la génération des paires de clés (privée/publique), la création et la signature des certificats, la gestion des requêtes de signature (CSR), la révocation via les CRL (Certificate Revocation List) et la vérification de la validité des certificats.

L'ensemble de ces fonctionnalités est accessible à travers une interface web développée avec Flask, qui permet une gestion centralisée, sécurisée et simplifiée de l'infrastructure.

0.4 Fonctionnalités attendues

Dans le cadre de ce projet, plusieurs fonctionnalités sont attendues pour assurer une gestion complète et sécurisée de l'infrastructure PKI. Ces fonctionnalités couvrent l'ensemble du cycle de vie des certificats, de leur création à leur révocation :

- Création de paires de clés RSA ou ECDSA pour chaque entité (Root CA, Intermediate CA, utilisateurs).
- Génération de certificats auto-signés pour l'Autorité Racine.
- Génération et gestion des CSR (Certificate Signing Requests).
- Signature des certificats des entités finales par l'Autorité Intermédiaire.
- Stockage organisé des certificats, clés privées/publiques, fichiers de configuration et métadonnées (fichiers index, numéros de série, etc.).
- Publication de listes de révocation (CRL) en cas de compromission ou d'expiration d'un certificat.
- Vérification de la validité et de la chaîne de confiance des certificats.
- Mise en place d'une interface web (Flask) pour gérer facilement la génération, la signature, la révocation et l'affichage des certificats.

0.5 Contraintes techniques et recommandations

Pour garantir une implémentation efficace et sécurisée de la PKI, plusieurs contraintes techniques doivent être respectées et des recommandations sont proposées :

- Utiliser OpenSSL en ligne de commande ou via des bibliothèques Python telles que `cryptography` ou `PyOpenSSL`.
- Respecter le standard X.509 pour la structure des certificats numériques.
- Sécuriser les clés privées par des permissions d'accès strictes et un stockage isolé.
- Maintenir une base de données pour les certificats émis (`index.txt`, `serial`, etc.).
- Générer des numéros de série uniques pour chaque certificat afin d'assurer une traçabilité.
- Mettre en place une procédure de révocation rigoureuse via les CRL (Certificate Revocation Lists).
- Documenter toutes les étapes de l'implémentation pour assurer la reproductibilité du système.
- Déployer l'interface web sur un système Linux sécurisé (ex. : Ubuntu).
- Vérifier régulièrement la chaîne de confiance pour prévenir les failles de sécurité.
- Utiliser un IDE moderne (ex. : VS Code) pour faciliter le développement et la maintenance du code.

0.6 Bibliothèques et outils utilisés

Dans le cadre de ce projet, plusieurs bibliothèques Python ainsi que des outils en ligne de commande ont été utilisés pour assurer la gestion des certificats numériques, la sécurité, et le développement de l'interface utilisateur. Voici la liste des bibliothèques utilisées avec leurs versions respectives :

- **Flask (v2.3.2)** :
Framework web léger permettant de créer une interface utilisateur simplifiée pour gérer les différentes fonctionnalités de la PKI. Il est utilisé pour :
 - Le routage des pages web (accueil, soumission de CSR, téléchargement de certificats),
 - Le rendu des templates HTML,
 - L'interaction avec la base de données et les fichiers système.
- **Werkzeug (v2.3.6)** :
Outil bas niveau utilisé par Flask pour la gestion des requêtes HTTP, la sécurisation des sessions et le routage. Il renforce la structure interne de l'application Flask.

- **cryptography (v41.0.3) :**

Bibliothèque puissante et sécurisée permettant la génération de clés cryptographiques, la création et la signature de certificats X.509, la gestion des CSR (Certificate Signing Requests), ainsi que les CRL (Certificate Revocation Lists).

- **python-dotenv (v1.0.0) :**

Utilisée pour charger les variables d'environnement à partir d'un fichier `.env`. Elle permet de sécuriser la configuration de l'application (ex : chemins des répertoires, mots de passe, configurations sensibles).

- **sqlite3** (intégrée à Python) :

Permet de stocker les informations concernant les certificats émis, les utilisateurs, les numéros de série, etc. C'est une solution légère et suffisante pour un projet académique.

Autres outils essentiels :

- **OpenSSL :**

Utilisé en ligne de commande pour créer des certificats auto-signés, générer des CSR, signer les certificats finaux et produire les CRL. Il s'appuie sur les standards de la norme X.509.

- **Invite de commandes Windows (CMD) :**

Utilisée comme terminal principal pour exécuter les commandes OpenSSL, lancer le serveur Flask et interagir avec les fichiers de l'infrastructure.

0.7 Interface Utilisateur Web (Flask)

L'interface utilisateur développée avec Flask vise à simplifier l'interaction avec l'Infrastructure à Clés Publiques (PKI). Elle permet aux utilisateurs d'effectuer facilement les opérations essentielles liées à la gestion des certificats numériques, tout en offrant une expérience utilisateur claire et intuitive.

0.7.1 Fonctionnalités principales de l'interface

- **Accueil (Home) :**

Page d'introduction présentant brièvement la PKI, ses objectifs, et les actions

disponibles à partir de l'interface.

- **Soumission de CSR (Certificate Signing Request) :**

Un formulaire permettant aux utilisateurs de téléverser un fichier `.csr` pour demander la signature d'un certificat par l'autorité intermédiaire. L'utilisateur peut également remplir ses informations (nom, email, etc.) selon le formulaire choisi.

- **Signature et génération de certificats :**

Une fois le CSR reçu, l'interface permet à l'administrateur de signer le certificat via l'autorité intermédiaire et de générer un fichier `.crt` ou `.pem` à destination de l'utilisateur.

- **Téléchargement de certificats :**

L'utilisateur peut accéder à la liste des certificats signés et télécharger le sien à travers l'interface, avec des informations associées (numéro de série, date de validité, etc.).

- **Révocation de certificats :**

Une interface permettant à l'administrateur de révoquer un certificat (par exemple en cas de compromission) et de générer une nouvelle CRL (Certificate Revocation List).

- **Vérification de certificats :**

Une page permettant à l'utilisateur de téléverser un certificat pour vérifier sa validité, sa chaîne de confiance, ou sa révocation.

0.7.2 Avantages de l'interface

- Réduction des erreurs manuelles dans les commandes OpenSSL,
- Accessibilité via navigateur web sans passer par la ligne de commande,
- Interface intuitive adaptée aussi bien aux débutants qu'aux administrateurs systèmes.

0.7.3 Structure générale de l'interface

L'application web se compose de plusieurs routes principales qui correspondent aux actions que peut effectuer un utilisateur ou un administrateur :

- **Page d'accueil (/) :** Cette page présente l'objectif de la plateforme, explique brièvement ce qu'est une PKI, et propose les liens vers les principales fonctionnalités

de l'application.

- **Soumission d'une requête de signature (/submit-csr)** : Cette page contient un formulaire permettant de télécharger un fichier CSR (Certificate Signing Request). Le serveur traite ensuite cette demande et génère un certificat signé si le fichier est valide.
- **Liste des certificats disponibles (/certificates)** : Affiche tous les certificats qui ont été signés par la CA intermédiaire. Chaque certificat peut être consulté ou téléchargé par l'utilisateur.
- **Révocation de certificat (/revoke)** : Permet à l'administrateur de révoquer un certificat donné via son numéro de série. Une fois révoqué, le certificat est ajouté à la CRL (Certificate Revocation List).
- **Vérification d'un certificat (/verify)** : Propose un formulaire pour télécharger un certificat à vérifier. L'application teste la chaîne de confiance et la validité du certificat (non expiré, non révoqué).
- **Téléchargement de la CRL (/crl)** : Permet d'obtenir le fichier CRL actualisé contenant la liste des certificats révoqués.

Objectifs de l'interface

- Simplifier l'utilisation des outils PKI sans ligne de commande.
- Fournir un point d'accès unique et centralisé pour toutes les opérations.
- Rendre la PKI compréhensible et accessible aux utilisateurs non techniques.
- S'assurer de la traçabilité des opérations (stockage dans une base de données SQLite).

Avantages de cette approche

- Aucune manipulation directe d'OpenSSL n'est requise par l'utilisateur final.
- L'automatisation des processus réduit le risque d'erreurs humaines.
- Une interface graphique améliore l'ergonomie et la compréhension du fonctionnement de la PKI.

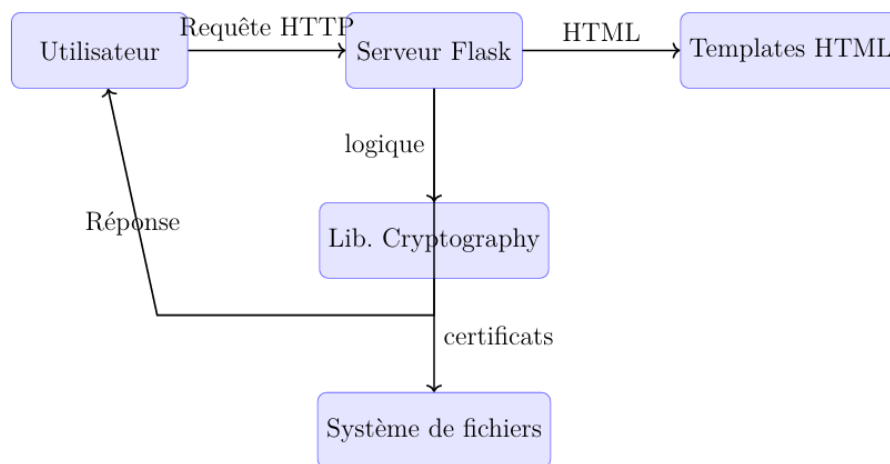
Remarque : Chaque fonctionnalité pourra être accompagnée d'une capture d'écran correspondante dans le rapport final pour une meilleure visualisation.

0.8 3.2 Architecture générale de l'application

L'application adopte une architecture MVC (Modèle-Vue-Contrôleur) simplifiée, adaptée au micro-framework Flask, permettant une séparation claire des responsabilités :

- **Modèle** : gestion des fichiers liés à la sécurité, tels que les clés cryptographiques, les certificats numériques et les listes de révocation de certificats (CRL).
- **Vue (templates)** : interfaces utilisateur dynamiques réalisées avec les templates Jinja2, facilitant la génération de pages HTML interactives.
- **Contrôleur (routes Flask)** : logique métier implémentée via les routes Flask, assurant des fonctionnalités telles que la génération, la signature et la vérification des certificats.

Schéma global du fonctionnement :



0.8.1 Diagramme de classes (modèle métier)

Le modèle métier est basé sur des entités représentées par des fichiers ou objets manipulés :

- **CSR** : contient le nom commun, les données brutes et la méthode d'enregistrement.
- **Certificat** : nom, date de validité, statut de révocation, avec fonctions de signature et de révocation.
- **CA** : contient les clés, et la capacité à signer un CSR.

Relations :

- Le **CSR** est signé par la **CA**.

- La **CA** émet un **Certificat**.
- Le **Certificat** peut être inscrit dans une **CRL**.

0.8.2 Modèle de données

L'application n'utilise pas de base de données relationnelle. Les données sont organisées selon une structure de fichiers hiérarchique :

- `pki/intermediateCA/` : contient la clé et le certificat de la CA intermédiaire
- `pki/final-certs/` : stocke les clés privées, CSR et certificats finaux
- `pki/crl/` : contient les fichiers CRL (au format PEM)

Exemple de structure :

```
pki/
  intermediateCA/
    intermediate.cert.pem
    intermediate.key.pem
  final-certs/
    exemple.csr.pem
    exemple.crt.pem
    exemple.key.pem
  crl/
    intermediate.crl.pem
```

Cette organisation facilite la portabilité, la lisibilité et la manipulation des opérations cryptographiques sans dépendre d'une base de données externe.

0.9 Implémentation du Projet

Ce chapitre présente en détail les principales fonctionnalités implémentées dans l'application web. Chaque fonctionnalité correspond à une étape importante de la gestion d'une infrastructure PKI.

0.9.1 Génération de la clé privée et CSR

L'utilisateur peut générer une paire de clés RSA (2048 bits) ainsi qu'une demande de certificat (CSR — *Certificate Signing Request*) via un formulaire web. La clé privée est enregistrée localement dans le dossier `final-certs/`.

- Le CSR contient la clé publique et les informations d'identité.
- Il est enregistré au format PEM.

4.2 Signature par l'autorité intermédiaire

Les demandes CSR générées peuvent être signées par la CA intermédiaire afin de produire des certificats finaux valides.

- La CA charge son certificat et sa clé privée.
- Le CSR est vérifié, puis signé.
- Le certificat contient la durée de validité (365 jours par défaut) et les extensions nécessaires.

Révocation et gestion de la CRL

L'application permet de révoquer un certificat et d'ajouter son numéro de série à la CRL (Certificate Revocation List).

- Les certificats révoqués sont chargés à partir de la CRL existante.
- Une nouvelle entrée est ajoutée avec la date de révocation et une raison (ex : `key_compromise`).
- La CRL est ensuite signée par la CA et sauvegardée.

Les listes du certificats

L'application fournit un tableau de bord interactif pour la supervision de l'infrastructure PKI :

- Nombre de certificats signés
- Nombre de certificats valides
- Suppression des certificats
- Révocation des certificats

0.10 Captures d'écran de l'interface utilisateur

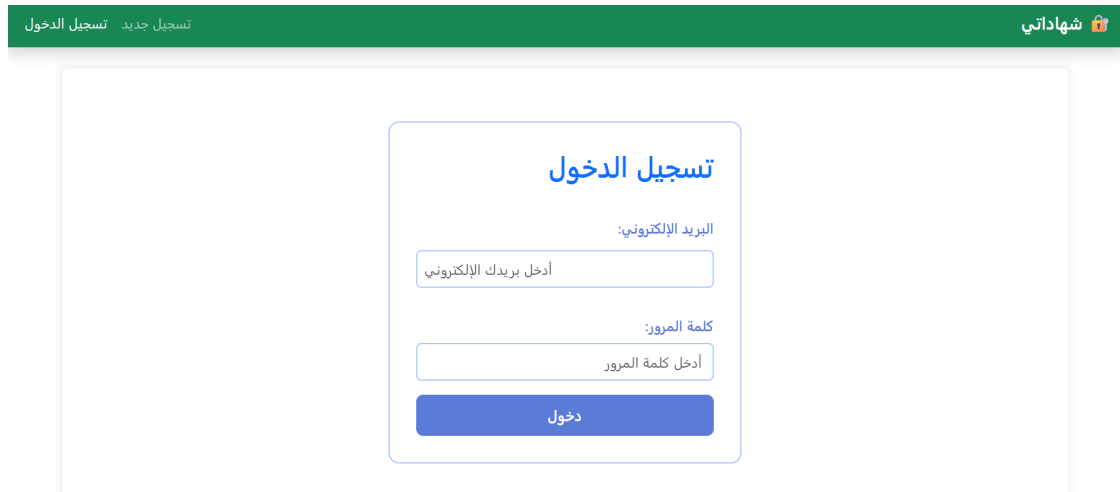


Figure 1– Se connecter

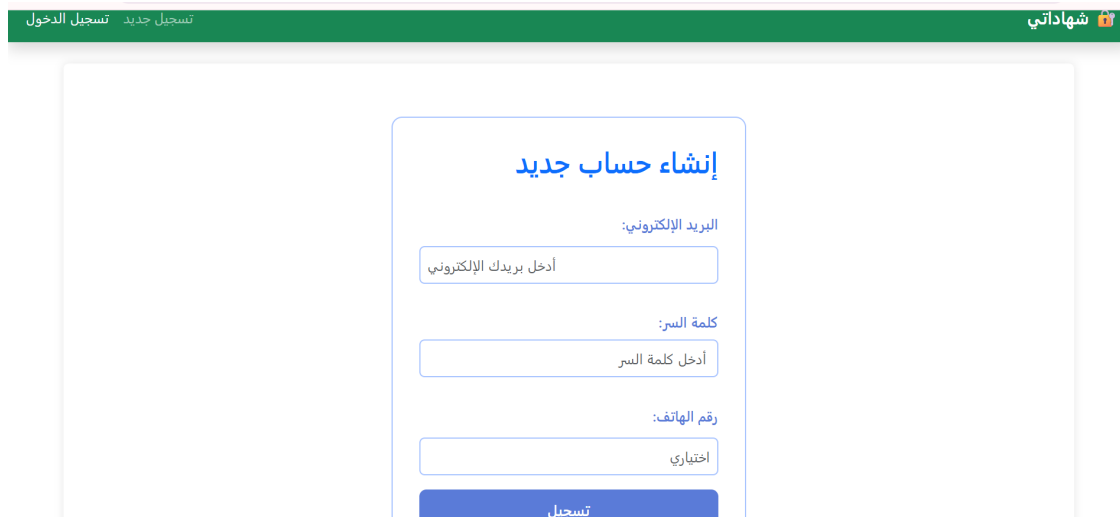


Figure 2–Créer un nouveau compte

توليد طلب توقيع شهادة (CSR)

الاسم الشائع (Common Name):

المنظمة (Organization):

الوحدة التنظيمية (Organizational Unit):

الدولة (Country):

البريد الإلكتروني (Email):

توليد

Figure 3– Générer une demande de signature de certificat

شهاداتي

توليد CSR توقيع CSR رفع شهادة الشهادات تسجيل الخروج

توقيع شهادة

رفع ملف طلب شهادة CSR لتوقيعه:

No file chosen Choose File

توقيع الشهادة ✓

تحميل طلب توقيع الشهادة (CSR)

العودة للصفحة الرئيسية

نظام إدارة البنية التحتية للمفاتيح العامة © 2025 PKI Manager

Figure 4– Signer un certificat

شهاداتي

توليد CSR توقيع CSR رفع شهادة الشهادات تسجيل الخروج

رفع شهادة جديدة

اختر ملف الشهادة (pem.):

No file chosen Choose File

رفع الشهادة ✓

تأكد أن الملف بصيغة PEM ويحتوي على شهادة صحيحة.

نظام إدارة البنية التحتية للمفاتيح العامة © 2025 PKI Manager

Figure 5– Importer un nouveau certificat

توليد CSR توقيع CSR رفع شهادة الشهادات تسجيل الخروج						
ائمة الشهادات						
إجراءات	الحالة	إلى	من	الرقم التسلسلي	المصدر	
حذف إلغاء	فعالة	2026-05-27 16:21:08	2025-05-27 16:21:08	710270584599618913243478332399564075311908970618	contact@mypki.local,CN=intermediate-= 1.2.840.113549.1.9.1 ca.mypki.local,OU=Intermediate CA,O=MyPKI,ST=Casablanca-Settat,C=MA	tanger2@gn
حذف إلغاء	فعالة	2026-05-27 17:03:05	2025-05-27 17:03:05	21282986531975491289805680795394285212484823247	contact@mypki.local,CN=intermediate-= 1.2.840.113549.1.9.1 ca.mypki.local,OU=Intermediate CA,O=MyPKI,ST=Casablanca-Settat,C=MA	Tanger1@gm
حذف إلغاء	فعالة	2026-05-27 17:06:22	2025-05-27 17:06:22	231325937158182636260174485706295233700722603548	contact@mypki.local,CN=intermediate-= 1.2.840.113549.1.9.1 ca.mypki.local,OU=Intermediate CA,O=MyPKI,ST=Casablanca-Settat,C=MA	tanger2@gmail

Figure 6– Liste des certificats

0.11 Conclusion

La mise en place d'une Infrastructure à Clés Publiques (PKI) à trois niveaux m'a permis de consolider mes connaissances en cryptographie appliquée et en gestion des certificats numériques. Ce projet a démontré l'importance d'une architecture sécurisée pour garantir la confiance et l'intégrité des échanges dans un environnement numérique.

Grâce à l'interface web développée, la gestion du cycle de vie des certificats — de la génération à la révocation — est rendue plus accessible aux administrateurs. Le respect des standards internationaux tels que X.509 renforce l'interopérabilité et la fiabilité de l'infrastructure.

En conclusion, cette expérience pratique offre une vision concrète des enjeux liés à la sécurité numérique et constitue un socle essentiel pour le développement futur de systèmes sécurisés.