REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTRE DE L'ENSEIGNEMENT SUPERIEUR

ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE SAAD DAHLEB BLIDA

Faculté des sciences
Département : Informatique

Mémoire

Pour l'obtention du diplôme de

Master

En informatique

Option : sécurité des systèmes d'information

Présenté par :

REBHAOUI AYOUB & HEMMADA MOHAMED ISLEM

THEME

Mise en place d'un système cryptographique hybride basé sur l'infrastructure de gestion des clés IGC/PKI

Devant le jury composé de :

- Mme MEZZI Melyara	USDB	Promotrice
- Mme SEMAR-BITAH Kahina	CDTA	Encadreur
- Mme BOUDERBALA Fatma Zohra	CDTA	CO-Encadreur
- Mme GHEBGHOUB	USDB	Président
- Mme HADJ HENNI	USDB	Examinateur

Année Universitaire 2020/2021

REMERCIMENT :

Avant tout nous tenons à remercier le bon Dieu le tout puissant, le très miséricordieux qui nous a donné la force et le courage de réaliser ce modeste travail. Nous tenons très sincèrement à remercier mes très chers parents dont leur soutien et le conseil nous ont toujours affiché une clairvoyance de la vie.

En préambule à ce mémoire, il m'est agréable de citer et adresser mes remerciements les plus sincères aux personnes qui m'ont apporté leurs aides et qui ont contribué à l'élaboration et au bon déroulement de ce travail :

A ma promotrice Mme **MEZZI Malyara**

A mon encadreur Mme **SEMAR Kahina**

Je tiens aussi à remercier A mon co-encardreur Mme **BOUDERBALA Fatma Zohra** .

J'ai remercions également tous ceux et celles qui ont participé de près ou de loin à l'aboutissement de cette concrétisation

À tout le corps enseignants et le personnel du département génie informatique qui ont contribués de près ou de loin à ma formation.

Aux membres de jury qui auront à juger et ce travail et d'avoir accepté de l'examiner.

DEDICACE :

Je dédie ce travail à :

Je dédie ce modeste travail aux êtres qui me sont les plus chers, je cite : Les parent les plus chers au monde, que dieu les garde et les protégés.

Tous ceux qui portent le nom « HEMMEDA » et «REBHAOUI»

A Mongi

A tous nous Amis sans exception

Résumé

La cryptographie hybride est un système de cryptographie faisant appel aux deux grandes familles de systèmes cryptographiques : la cryptographie asymétrique et la cryptographie symétrique. Le présent travail propose une solution de cryptographie hybride basée sur les PKI pour le Centre de Développement des Technologies Avancées garantissant l'intégrité et la confidentialité des documents échangés entre les utilisateurs de la plateforme ‘cdta’ . En premier lieu, on a présenté le domaine de la cryptographie. En second, on a fait le choix d’intégrer un provider de sécurité afin d’utiliser ‘hybride’ et fournir les algorithmes de cryptographie, où nous avons choisi l’algorithme AES pour le cryptage symétrique et l’algorithme RSA pour le cryptage asymétrique en basant sur openssl pour l’utilisation des certificats. En dernier la solution a été mis en place.

abstract

Hybrid cryptography is a cryptography system using two main families of cryptographic systems: asymmetric cryptography and symmetric cryptography. This work proposes a hybrid cryptography solution based on PKIs for the Advanced Technologies Development Center guaranteeing the integrity and confidentiality of documents exchanged between users of the ‘cdta’ platform. First, the field of cryptography was presented. Second, we made the choice to integrate a security provider in order to use 'hybrid' and provide the cryptography algorithms, where we chose the AES algorithm for symmetric encryption and the RSA algorithm for asymmetric encryption. based on openssl for certificate usage. Lastly the solution was implemented.

ملخص

التشفير الهجين هو نظام تشفير يستخدم مجموعتين رئيسيتين من أنظمة التشفير: التشفير غير المتماثل والتشفير المتماثل. يقترح هذا العمل حل تشفير هجين يعتمد على البنية التحتية للمفتاح لمركز تطوير التقنيات المتقدمة ، مما يضمن سلامة وسرية المستندات المتبادلة بين مستخدمين منصة الشركة . أولاً ، تم تقديم مجال التشفير. ثانيًا ، قمنا باختيار دمج موفر أمان من أجل الاستخدام 'هجين' وتقديم خوارزميات التشفير ، حيث اخترنا خوارزمية للتشفير المتماثل وخوارزمية للتشفير غير المتماثل. على أساس ' اوبن سسل ' لاستخدام الشهادة. وأخيرا تم تنفيذ الحل .

CHAPITRE III: Analyse et conception

CHAPITRE IV : Réalisation

Liste des figures

Liste des tableaus

Introduction générale

Plus on avance dans le temps, plus les menaces se multiplient et se diversifient, aussi le risque de compromission de fichiers ou de messages est de plus en plus important. D'où l'importance de la cryptographie, dont la mission, vue de manière très large, est que seul le destinataire initial d'un message ou d'un document puisse en prendre connaissance. En cas de fuite de données par exemple. Le centre de développement de technologies avancées comme toute entreprise moderne a envisagé de sécuriser son système d'information. Ceci devient essentiel et peut éviter les conséquences parfois dramatiques pour l'entreprise.

Le travail demandé consiste a développé un module de chiffrage des fichiers manipulés entre les différents intervenant, en utilisant les algorithmes et les technique moderne de chiffrement. Il est demandé de mettre en place un serveur de PKI pour garantir l'identité de chaque entité intervenante. ainsi d'aller vers le cryptage hybride pour ses avantage.

Le mémoire est organisé autour de quatre chapitres :

Le premier chapitre est une introduction à la cryptographie. Il aborde entre autres les deux principaux schémas de chiffrement en cryptographie qui sont le chiffrement à clé secrète et le chiffrement à clé publique, ainsi que les fonctions de hachages cryptographiques.

Le deuxième chapitre présente une vue sur les infrastructures de gestion de clé et l'architecture de serveur PKI.

Le troisième chapitre traite l'analyse des besoins exprimé par le CDTA et la conception de la solution que nous avons proposée.

Le quatrième chapitre regroupe les outils utilisés pour la réalisation de ce système et quelques interfaces de notre application.

Enfin, notre travail est conclu par une conclusion générale et quelques perspectives ouvertes qui peuvent être envisagées comme suite à notre projet.

CHAPITRE I: Concepts et principes de la cryptographie

1 INTRODUCTION

La signature électronique sécurisée est étroitement liée aux technologies de la cryptographie à clé publique ou cryptographie asymétrique. La cryptographie permet de réaliser les objectifs de sécurité, de confidentialité, d'intégrité, d'authentification, et de disponibilité via ses outils de chiffrement. La connaissance de ces concepts est donc indispensable pour la mise en place de ce système de signature. Dans le présent chapitre, nous allons définir les concepts de baseet les principes essentiels dans le domaine de la cryptographie.

2 CONCEPTS DE BASE

2.1 La cryptologie

La cryptologie est un mot composé qui tire son origine du grec : cryptos qui signifie secret et logie qui signifie science. En fait, c'est la science du secret et ne peut être vraiment considérée ainsi que depuis peu de temps. Elle englobe la cryptographie, l'écriture secrète et la cryptanalyse, l'analyse de cette dernière[1] On peut dire que la cryptologie est un art ancien et une science nouvelle : un art ancien car Jules César l'utilisait déjà et il fait son apparition dans l'ancien testament sous la forme du code Atbash ; une science nouvelle parce que ce n'est que depuis les années 1970 qu'elle est devenue un thème de recherche scientifique. Cette discipline est liée à beaucoup d'autres, par exemple la théorie des nombres, l'algèbre, lathéorie de l'information, ou encore les codes correcteurs [2].

2.2 Les composants de la cryptologie

a- La cryptanalyse : La cryptanalyse est la technique qui consiste à déduire un texte en clair d'un texte chiffré sans possédé la clé de chiffrement. Le processus par lequel on tente de comprendre un message en particulier est appelé une attaque. Une attaque est souvent caractérisée par les données dont elle a besoin :

- Attaque sur texte chiffré seul (ciphertext-only en anglais) : Le cryptanalyste possède des exemplaires chiffrés des messages, il peut faire des hypothèses sur les messages originaux qu'il ne possède pas. Le cryptanalyse est plus ardue à cause de manque d'information à disposition.
- Attaque à texte clair connu (known-plaintext attack en anglais) : Le cryptanalyste possède des messages ou des parties de messages en clair ainsi que les versions chiffrées. La cryptanalyse linéaire fait partie de cette catégorie.
- Attaque à texte clair choisi (chosen-plaintext attack en anglais) : Le cryptanalyste possède des messages en clair, il peut créer les versions chiffrées de ces messages avec l'algorithme que l'on peut considérer comme boîte noire. La cryptanalyse différentielle est un exemple d'attaque à texte clair choisi.
- Attaque à texte chiffré choisi (chosen-ciphertext attack en anglais) : Le cryptanalyste possède des messages chiffrés et demande la version en clair de certains de ces messages pour mener l'attaque [2].

b- La cryptographie : est l'art de cacher l'information. Elle désigne l'ensemble des techniques permettant de chiffrer des messages, c'est-à-dire permettant de les rendre incompréhensibles [3]. Cela permet la protection de messages ou données en concevant des procédés ou algorithmes de chiffrement utilisant des secrets ou clés, assurant ainsi la confidentialité, l'authenticité et l'intégrité. Elle est utilisée depuis l'Antiquité, mais certaines de ses méthodes les plus importantes, comme la cryptographie asymétrique, datent de la fin du XXe siècle.

Les éléments constituant la cryptographie sont [2]:

➢ **Chiffrement ou cryptage**

Est un procédé de cryptographie grâce auquel on souhaite rendre la compréhension d'un document impossible à toute personne qui n'a pas la clé de déchiffrement.

➢ **Clé de chiffrement**

Paramètre constitué d'une séquence de symboles et utilisé, avec un algorithme cryptographique, pour transformer, valider, authentifier, chiffrer ou déchiffrer des données.

➢ **Déchiffrement**

C'est la méthode ou l'algorithme utilisé pour transformer un texte chiffré en un texte en clair.

➢ **Texte en clair**

C'est le message à protéger

➢ **Texte chiffré**

C'est le résultat du chiffrement du texte en clair.

➢ **Crypto-système**

C'est l'algorithme de chiffrement.

3 Historique de la cryptographie

3.1 Les premières méthodes de chiffrement (Antiquité)

Lors de l'Antiquité, la cryptographie était restreinte à un petit groupe de personnes qui avaient les capacités d'imaginer et de développer une telle idée (n'oublions qu'il n'y avait là aucun manuel ou aucune aide quelconque car la cryptographie n'existait pas encore vraiment). Ainsi, le risque que les messages codés ne soient découverts et décodés par d'autres personnes était faible. En effet, la plus grande partie des gens n'auraient même pas pu imaginer ce concept et même s'ils y avaient songé, les risques restaient minimes étant donné que la majorité de la population était illettrée.[3]

- **1900 av. J.-C.** Un scribe égyptien utilise des hiéroglyphes, qui ne sont pas standards, racontant la vie de son maître. Le but n'était pas de rendre le texte incompréhensible mais plutôt de lui donner un caractère plus solennel.
- **1600 av. J.-C.** Le premier document chiffré connu remonte à l'Antiquité. Il s'agit d'une tablette d'argile, retrouvée en Irak. Un potier y avait gravé sa recette secrète en supprimant des consonnes et en modifiant l'orthographe des mots.
- **600 av. J.-C.** Un roi de Babylone écrit sur le crâne rasé de ses esclaves, attend que leurs cheveux aient repoussé, et les envoie à ses généraux. Il suffit ensuite de raser à nouveau le messager pour lire le texte.
- **600 av. J.-C.** La Mésopotamie, grande civilisation de l'antiquité, avait atteint un niveau cryptologique étonnamment moderne. On a retrouvé en Iran des fragments de tablettes où des nombres correspondaient à des mots.
- **500 av. J.-C**. Des scribes hébreux emploient le ATBASH, un simple algorithme de chiffrement par substitution utilisant l'alphabet renversé, afin de transcrire le livre de Jeremiah. (Par exemple bonjour devient _ ruojnob _).
- **487 av. J.-C.** Des grecs utilisent une scytale, aussi appelé bâton de Plutarque (historien et moraliste de la Grèce Antique). Il s'agit d'un bâton autour duquel on enroulait une longue et mince bande de cuir sur laquelle on écrivait notre message secret (souvent codé). Une fois la bande déroulée, il était difficile de retrouver le message. Seule le destinataire, connaissant le diamètre du bâton de celui ayant servi à écrire le message, pouvait le déchiffrer.

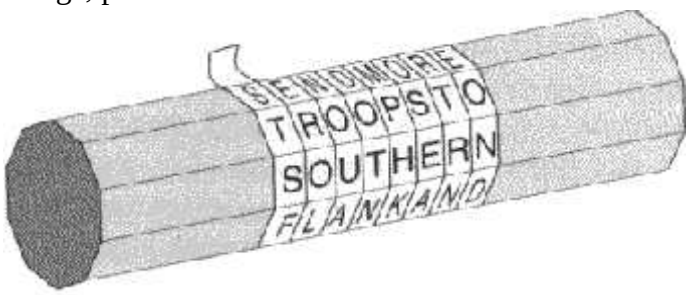

Figure 1:. Une Scytale

La principale faiblesse de ce système est qu'un bâton de diamètre approximativement égal suffit pour déchiffrer le texte.

3.2 Les premiers systèmes cryptographiques (période classique)

Dans le reste du monde la situation restait à peu près semblable jusqu'à environs 50 av. J.-C. L'homme prend enfin l'initiative de développer des techniques de protections de l'information confidentielle de façon plus efficace.

- **150 av. J.-C.** Polybe, historien grec, a imaginé un procédé de chiffrement très innovant pour son temps. Cette méthode utilise un système de transmission basé surun carré

de 25. Les spécialistes en cryptologie moderne ont vu dans cette méthode plusieurs caractéristiques très intéressantes dont la conversion de lettres en chiffres, la représentation de chaque lettre par deux éléments séparés.

- **50 av. J.-C.** Jules César a utilisé une substitution dans l'alphabet pour les communications gouvernementales. En effet, il décalait la lettre qu'il souhaitait coder de 3 lettres vers la droite (en revenant au début de l'alphabet si besoin). ''Le papyrus de Leyde'' est le plus ancien manuscrit connu concernant l'alchimie. Il utilise un algorithme de chiffrement pour cacher les parties importantes de certaines recettes.
- **1499** Jean Trithème (1462-1516), ancien abbé, est considéré comme un des pères de la cryptographie. En effet, il est l'auteur d'un des premiers systèmes poly-alphabétiques et il a créé une technique de sténographie (fait de cacher un message au sein d'un autre) où les lettres sont remplacées par des mots choisis de manière à former, par leur réunion, une prière par exemple.
- **1560** Blaise de Vigenère (1523 - 1596) est l'auteur de l'un des premiers systèmes de substitution poly-alphabétique, il a utilisé donc une clé. Cette méthode restera dominante pendant trois siècles. Sa particularité est qu'il n'utilise non pas un alphabet, mais 26 alphabets décalés pour chiffrer un message. On peut résumer ces décalages avec un carré de Vigenère. Ce chiffre utilise une clé qui définit le décalage pour chaque lettre du message [4].

3.3 La Cryptographie moderne (Temps Moderne)

L'éducation, l'accès à l'information et la connaissance sont devenus accessibles à presque tout le monde, avantageant énormément la créativité et la nouveauté et donc la complexité. Les méthodes de cryptographies se sont décuplées et la difficulté de
cassage du code également. Cependant, la cryptographie a évolué uniquement dans desmilieux fermés tels, les gouvernements, les services secrets ou les armées. C'est pourquoi, pendant tant d'années, elle est restée une science secrète.

- **1918** Gilbert Vernam met au point l'algorithme ''One Time Pad'' (traduit masque jetable) aussi appelé chiffre de Vernam. En effet, il fonctionne sur le même principe que le chiffrement de Vigenère, avec quelques règles supplémentaires : Une clé ne doit être utilisée qu'une seule fois, elle doit être de la même taille que le message et elle doit être générée aléatoirement. Ce système est donc reconnu comme étant l'algorithme de chiffrement le plus sécuritaire. Cependant, la communication des clés pose problème car n'oublions pas que la clé doit être de la même taille que le message codé, il est donc hors de question d'utiliser Internet comme passerelle. Pour donner un exemple, on peut penser à la valise diplomatique que les gouvernements utilisent pour communiquer les clés privées de façon sûre à leurs ambassades.

- **1923** Le Dr Arthur Scherbius, hollandais résidant en Allemagne, met au point une machine nommée Enigma qui sert à encoder des messages. Pendant la guerre, des versions d'Enigma sont utilisées pour pratiquement toutes les communications radio allemandes ainsi que pour les communications télégraphiques. Même les bulletins météos sont codés avec Enigma. Elle continuera à être utilisée dans l'armée encore jusqu'en 1939 (date à laquelle le code de cette machine a été cassé).

Figure 2 : la machine Enigma [10]

- **1976** IBM publie un algorithme basé sur Lucifer (l'une des premières méthodes de chiffrement moderne destiné à un usage civil). Il devient le DES (Data Encryption Standard). C'est un chiffrement qui transforme des blocs de 64 bits avec une clé secrète de 56 bits au moyen de permutations et de substitutions. Le DES est considéré comme étant raisonnablement sécuritaire.
- **1978** Le RSA est inventé par Ronald L. Rivest, Adi Shamir et Leonard M.Adleman. Il est un des plus populaires des systèmes à clés publiques.
- **1992** Le MD5 (Message Digest 5) est développé par Ronald L. Rivest. Il s'agit d'une fonction de hachage très utilisée sur l'Internet, mais n'est pas considéré comme étant un algorithme sûr.
- **2000** L'AES (évolution de l'algorithme Rijndael) devient le standard du chiffrement avancé pour les organisations du gouvernement des Etats-Unis.

- **2005** La cryptographie quantique, qui repose sur la physique quantique, serait considérée comme sûre à presque 100%. Ce sera peut-être la méthode du futur car elle est toujours en cours d'expérimentation[6]

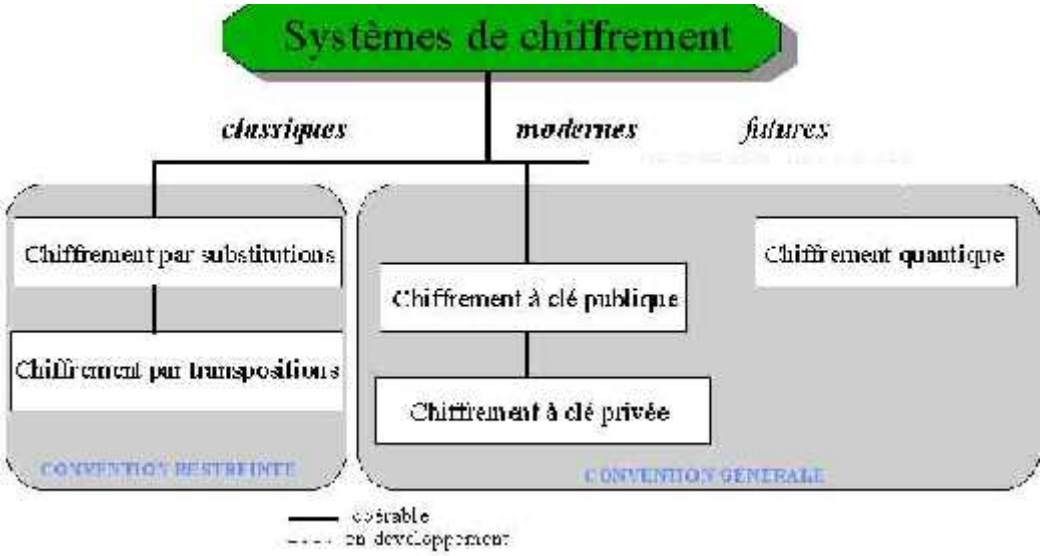

Figure 3 :Les systèmes de chiffrement[5]

4 Fonctionnement de la cryptographie moderne

À côté de la fonction de chiffrement, qui permet de préserver le secret des données lors d'une transmission, et qui a été utilisée depuis très longtemps, la cryptographie moderne a développé de nouveaux buts à atteindre et qu'on peut énumérer de manière non exhaustive: confidentialité, intégrité des données, authentification des divers acteurs, non-répudiation d'un contrat numérique, signature numérique, certification, contrôle d'accès, gestion des clés, preuve de connaissance.

La cryptologie moderne a pour l'objet l'étude des méthodes qui permettent d'assurer les services d'intégrité, d'authenticité et de confidentialité dans les systèmes d'information et de communication. Elle recouvre aujourd'hui également l'ensemble des procédés informatiques devant résister à des adversaires[7].

4.1 La cryptographie symétrique

Le principe de la cryptographie symétrique, encore appelée cryptographie conventionnelle ou à clé secrète est d'utiliser la même clé pour chiffrer et déchiffrer l'information. L'avantage de ce type de cryptographie est la rapidité des processus de chiffrement et déchiffrement. C'est pour cette raison qu'elle est largement utilisée pour protéger des données de taille importante. Cependant, utiliser la cryptographie conventionnelle pour la transmission des messages peut

rapidement revenir très cher, à cause de la difficulté de partager la clé secrète avec un destinataire que l'on ne connaît pas ou avec qui l'on n'a aucun contact physique.

La cryptographie symétrique est donc recommandé pour le stockage de données car le chiffrement est très rapide, mais il est vivement déconseillé de l'utiliser seule sans autre moyen de distribution de clés dans le cas de données à transmettre.

Les algorithmes de cryptographie conventionnelle les plus connus sont DES (Data Encryption Standard), 3DES, AES (Advanced Encryption Standard), RC4 (Ron's Code), RC5[8].
le DES (Data Encryption Standard) est L'algorithme symétrique le plus célèbre qui fonctionnait avec des clés de 64bits remplacé par l'AES (Advanced Encryption System, qui fonctionne avec des clé allant jusqu'à 256 bits)[1].

Figure 4: Principe du chiffrement symétrique

La limitation de l'utilisation de la cryptographie symétrique réside dans la problématique de la transmission en toute confidentialité de la clé secrète. Si le réseau est composé de n nœud alors il faudra gérer n.(n-1)/2 clés, ce qui ne s'adapte pas au facteur d'échelle. Avec 500 nœud, on arrive déjà à plus de 12 millions de clés à gérer[8]

Deux grandes catégories de systèmes de chiffrement à clés secrètes :.

- Le chiffrement par flux. [9]
- Le chiffrement par blocs.

4.1.1 **Le** chiffrement par flux (stream cipher)

Le chiffrement par flux est un chiffrement a clé symétrique permet de traiter des donnés de longueur quelconque. Il fonctionne en générant, à partir de la clé K, une suite de symboles, appelée suite chiffrante, de la même longueur que le message à chiffrer. Les bits du texte clair sont généralement combinés par opération XOR avec un flux de bits pseudo-aléatoire (*keystream*).Un des algorithmes de chiffrement par flot le plus rependu est RC4, il a été conçu en 1987 par Ronald Rivest [9]. Ces algorithmes sont généralement plus rapides mais moins résistants que les chiffrements par blocs.

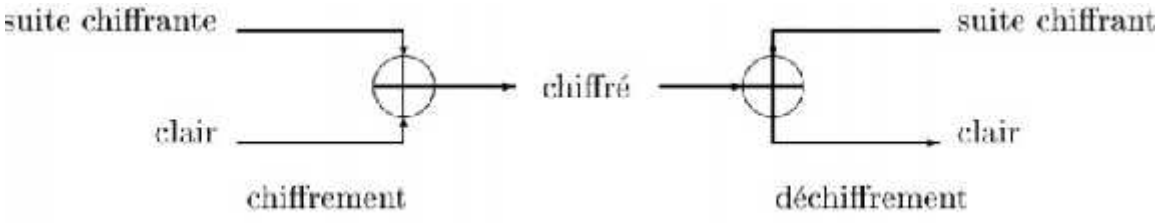

Figure 5: mécanisme de chiffrement par flot[8]

4.1.2 Le chiffrement par bloc (Block Chiper)

Le chiffrement par bloc permet de travailler sur des blocs de taille fixée. Le texte clair est préalablement découpé en "blocs de message" qui sont traités séparément. Comme la plupart des microprocesseurs traitent des mots de plusieurs bits (de 32 bits, souvent), cette opération s'avère rentable une fois mise en œuvre. Dans la pratique, on utilise des modes de chiffrement hybrides : on considère le texte clair comme un flot de blocs de message qui sont traités "au vol". On appelle "mode opératoire" ce mécanisme de traitement 'a partir d'une fonction de chiffrement par blocs[8].

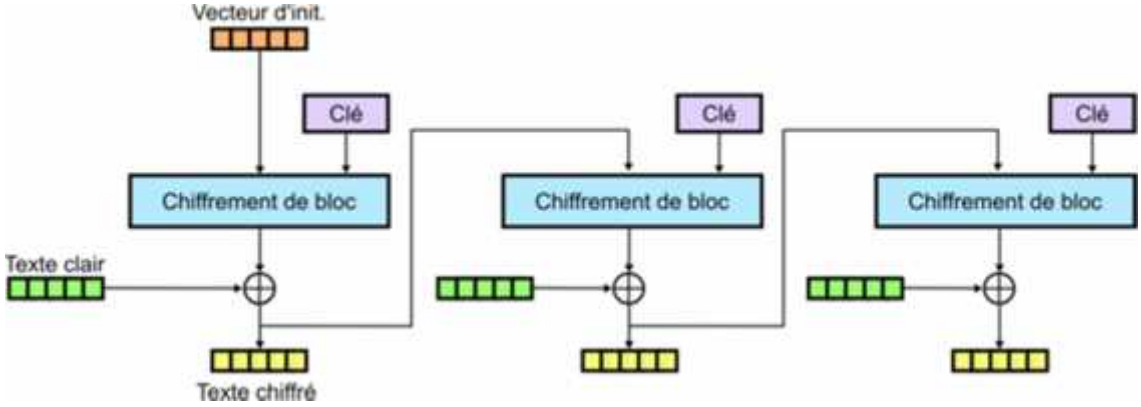

Figure 6: mécanisme de chiffrement par bloc

Un exemple de taille de bloc et de clé utilisés par les algorithmes les plus connus:

- DES : blocs de 64 bits, clé de 56 bits.
- IDEA : blocs de 64 bits, clé de 128 bits.
- AES : blocs de 128 bits, clé de 128 à 256 bits.

Pour cette catégorie, nous allons présenter deux algorithmes très connus DES et AES en mettant l'accent sur l'AES, car il est devenu le standard recommandé pour le chiffrement symétrique[11].

4.2 La cryptographie asymétrique

Le fonctionnement repose sur une paire de clés liées mathématiquement. On considère l'une d'entre elle comme privée, et l'autre comme publique (qui pourra donc être diffusée). Le contenu de la première clé ne peut être retrouvé à partir de la seconde clé. Les opérations se font à sens unique, on ne peut revenir en arrière (principe de la trappe). Ce retour en arrièreest toutefois possible en possédant la clé privée[12].

Un cryptosystéme à clé publique se comporte comme un coffre-fort dont seule une personne possède la clé. Il laisse son coffre ouvert à disposition de toute personne désirant lui envoyer un message, celle-ci referme lors la porte et seul le destinataire peut ensuite l'ouvrir. En pratique, le destinataire publie à l'intention de ceux qui veulent lui envoyer des messages. C'est une méthode de chiffrement que lui seul est capable de déchiffrer, on voit donc bien pourquoi ces systèmes sont dits asymétriques[16].

L'un des principaux avantages de la cryptographie de clé publique est qu'elle offre uneméthode d'utilisation des signatures numériques. Les algorithmes de cryptographie asymétrique les plus utilisés sont : Diffie-Hellman, RSA (Ron Rivest, Adi Shamir et Len Adleman), Elgamal, DSA (Digital Signature Algovrithm) [13].

Figure 7: Chiffrement Asymétrique [14]

Le problème posé dans cette méthode de cryptographie est comment garantir qu'une clé publique correspond bien à l'entité avec qui on communique ? si la clé publique n'est pas distribuée d'une manière sécurisée. Un schéma asymétrique peut subir une attaque de type "Man in the Middle", une telle attaque est illustrée dans le scénario ci-après :

Figure 8: Man in the middle[8]

La solution au problème dit "man in the middle" est l'usage d'un certificat numérique qui assure la liaison entre l'identité et la clé publique correspondante dans un document numérique signé par une tierce partie de confiance dite autorité de certification.

4.2.1 Les algorithmes de chiffrement asymétrique

Un chiffrement asymétrique est défini par trois algorithmes

- Algorithme de génération des clés,
- Algorithme de chiffrement,
- Algorithme de déchiffrement.

Certains algorithmes asymétriques comme RSA offrent aussi des opérations pour la génération de signature numérique et sa vérification.

L'utilisation d'une paire de clés publique/privée permet d'assurer la confidentialité, l'authentification, l'intégrité et l'échange de la clé secrète. Cependant, les algorithmes asymétriques ne réalisent pas tous ces fonctions[7]. La Table II.1 donne un résumé des opérations cryptographiques pouvant être réalisés par les algorithmes asymétriques les plus connus.

Tableau 1: Les applications des crypto systèmes asymétriques[5]

Algorithme	Chiffrement/Déchiffrement	Signature numérique	Echange de clé
RSA	Oui	Oui	Oui
Diffie-Hellman	Non	Non	Oui
DSA	Non	Oui	Non
EC-Elliptic Curves	Oui	Oui	Oui

a- Cryptage RSA :

Le cryptage RSA, du nom de ses concepteurs, Ron Rivest, Adi Shamir et Leonard Adleman, est le premier algorithme de chiffrement asymétrique. Il a été découvert en 1977 au Massachusetts Institute of Technologie. Le principe de ce cryptage est d'utiliser une clé publique pour crypter les données et une clé privée qui servira à les décrypter[7].

Les opérations de chiffrement et de déchiffrement dans le protocole RSA sont basées sur un calcul d'exponentiation modulaire, définies respectivement par les expressions : C =ME mod N (1) M =CD mod N (2) Où M est le message en clair, C est le message chiffré et N est le modulo. Ces deux opérations sont généralement exécutées après une étape de générations de deux clés. La première (E, N) est rendue publique. La seconde (D, N) est privée. Les deux exposants E et D sont calculés comme suit :

1. Génération deux nombres premiers p et q.

2. Calcul du modulo N, tel que $N=p\times q$.

3. Calcul de la fonction d'Euler: $\Phi(N)= (p-1) \times (q-1)$.

4. E est choisi, tel que 2 < E [8]

b- Protocole d'échange de clés DH (Diffie-Hellman) :

C'est un algorithme à clé publique d'échange de clé, basé sur le problème de logarithme discret, développé par Diffie et Hellman en 1976. Ce protocole permet à deux tiers de générer un secret partagé sans avoir aucune information préalable l'un sur l'autre et en échangeant uniquement leurs clés publiques[7].

c- DSA (Digital Signature Algorithm):

L'algorithme de Diffie-Hellman permet de créer un secret commun, mais contrairement à RSA, il ne permet pas signer des documents. C'est pour cette raison que l'algorithme de Diffie-Hellman est souvent associé à DSS (Digital Signature Standard, un autre algorithme)ou DSA (DSS permet de signer les documents). Le DSA est une norme promulguée par le NIST. Il est uniquement utilisé pour les signatures numériques. DSA génère des signatures plus rapides, et peut vérifier les signatures RSA[7].

4.3 La cryptographie hybride

Une solution peut d'utiliser les deux systèmes de cryptographie et de prendre l'avantage de chacun. C'est ce qu'on appelle la cryptographie hybride, elle utilise la cryptographie à clé publique pour échanger la clé secrète qui va être utilisée pour le chiffrement des données. Cela a l'avantage de protéger la clé secrète et d'être rapide car c'est la cryptographie symétrique qui va être utilisée pour le chiffrement [4].

5 Notion de hachage

Une fonction de hachage est une fonction qui transforme en un résumé court, de taille fixe. L'image d'un message par une fonction de hachage s'appelle le condensé du message, L'empreinte du message, le résumé du message ou encore le message haché. Une fonction de hachage doit posséder deux qualités indispensables:

_ Résistance à la détermination d'une primage, ce qui signifie qu'il doit être impossible en pratique, à partir d'un résumé m, de trouver un message M ayant ce résumé tel que m=h(M).

_ Résistance aux collisions, ce qui signifie qu'il est impossible en pratique de construire deux messages M1 et M2 ayant le même résumé : h(M1)=h(M2) [5].

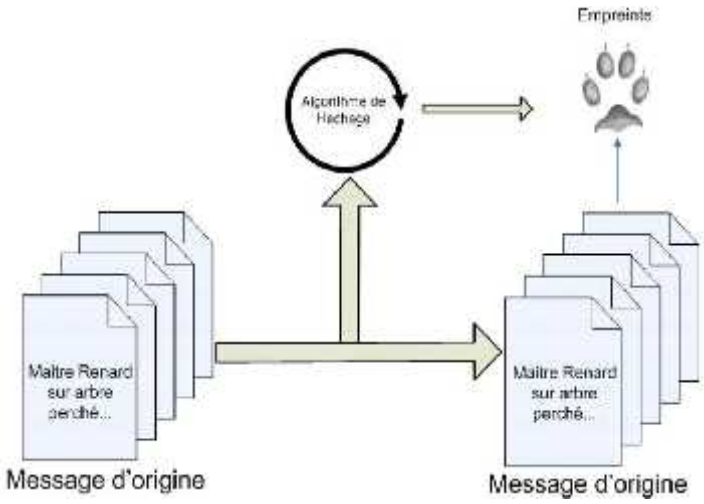

Figure 9: fonctionnement de hachage [6]

Les algorithmes de hachage les plus utilisés sont :

- **MD4** : Message Digest 4 (RFC 1320) qui génère une empreinte de 12 bits. Cet algorithme est désormais abandonné à cause d'une faiblesse de conception et tant la probabilité d'avoir le même résultat pour deux messages différents est important.
- **MD5** : Message Digest 5 (RFC 1321) améliore MD4, l'empreinte reste sur 128 bits. Mais, cet algorithme est désormais considéré comme non sûr pour l'usage cryptographique : une équipe de Chinois a pu démontrer pouvoir reproduire à partir d'une empreinte e1 (calculée à partir d'un message X), un nouveau message (Y) capable de produire une empreinte e2 identique à e1. Ce qu'on appelle une collision complète (dans un contexte peu sécurisé puisque le second message n'a pas été trouvé de manière aléatoire…). Il n'empêche qu'il est encore fortement utilisé pour vérifier l'empreinte de fichiers téléchargés sur Internet, pour le cryptage de mot de passe surles sites Web (fonctions PHP disponibles).
- **SHA-0 :** Secure Hash Algorithm 0, cette première version mise au point en 1993 fut vite abandonnée à cause de deux failles permettant des collisions.
- **SHA-1** : Secure Hash Algorithm 1 est sorti en 1995 sous la coupelle de la NSA, il produit une empreinte de 160 bits, mais reste parmi les algorithmes peu sûrs, bien qu'il possède une puissance de calcul importante pour permettre une collision qui ne sera trouvée à partir d'un message aléatoire.
- **SHA-2** (aussi appelé SHA-224, SHA-256, SHA-384, SHA-512) Secure Hash Algorithm C'est un algorithme dérivé de SHA1, publié par la NSA en 2000, les versions 256 et 512 sont répandues et dits sûrs, une étude de 2003 a montré que l'algorithme n'avait pas la fragilité des algorithmes rencontrés sur MD5 et SH1.
- **Whirlpool :** conçu dans le cadre d'un projet européen, cet algorithme génère des empreintes de 512 bits. La longueur de l'empreinte et le fait que l'algorithme travaille sur des registres 64 bits en font un système assez consommateur pour le processeur et la mémoire (notamment sur environnements embarqués et basés sur des architectures 32 bits), mais il reste particulièrement robuste, il est normalisé ISO 10118-3 en 2004 pour sa version finale et il est libre de droits.

D’autres algorithmes de hachage peuvent être rencontrés mais présentent pour la plupart : 1- une implémentation peu rencontrée, 2- des failles (collisions) mises en évidence. Citons notamment FFTH, RIPEMD, LANMAN Hash, Tiger [6].

6 Conclusion

Un concepteur de système cryptographique est toujours en train d’essayer d’élaborer un système de chiffrement plus sûr mais au même temps des intrus essayent de casser ce dernier, ils se livrent constamment une bataille, mais les enjeux sont toujours énormes.

Dans ce chapitre, nous avons défini les concepts de base de la cryptographie, les différents algorithmes de cryptage symétrique et asymétrique ainsi que les algorithmes de hachage. Dans le chapitre qui suit, on va définir la gestion de privilèges basée sur les certificats d'identité avec aperçu sur les infrastructures de gestion de clé publique ainsi qu’une étude sur les différents types et formats de la signature numérique.

CHAPITRE II: Généralités sur les infrastructures à clé publique

1-Introduction

Dans le présent chapitre, on montra l'intérêt des infrastructures de gestion de clé pour la gestion des privilèges basé sur les certificats d'identité. Ensuite, on présentera une étude descriptive surles différents standards de clé publique ainsi que le mécanisme et les différents formats de la signature numérique.

- Définition PKI (Public Key Infrastructure)

Est un système de gestion des clefs publiques qui permet de gérer des listes [18] importantes de clefs publiques et d'en assurer la fiabilité, pour des entités généralement dans un réseau. Elle offre un cadre global permettant d'installer des éléments de sécurité tels que la confidentialité, l'authentification, l'intégrité et la non-répudiation tant au sein de l'entreprise [19] que lors d'échanges d'information avec l'extérieur. Une infrastructure PKI fournit donc quatre services principaux:

- Fabrication de bi-clés.
- Certification de clé publique et publication de certificats.
- Révocation des certificats.
- Gestion la fonction de certification.

2-Les composants de l'infrastructure

L'infrastructure de gestion des clés est basée sur plusieurs composants qui sont indispensables à son fonctionnement. Parmi ces composants, nous répertorions comme principaux, les suivants.

2.2.1 – L'autorité de certification (CA)

On peut dire que c'est le composant le plus important de l'infrastructure PKI du fait de son rôle central dans les différentes cinématiques d'échanges à l'intérieur d'une PKI.

La CA est chargé de délivrer et gérer les certificats. En effet, elle génère des certificats à clés publiques et assure l'intégrité et l'authenticité des informations contenues en les signant avec sa clé privée. Pour émettre des certificats, elle doit recevoir, au préalable, les requêtes de certification contenant la clé publique de l'entité qui le sollicite.

2.2.2 – L'autorité d'enregistrement (RA)

Elle joue le rôle d'intermédiaire entre l'utilisateur et la CA et dépend de cette dernière. Elle a comme responsabilité de vérifier tout ce qui concerne l'utilisateur, son identité, la concordance entre clés privées/publiques, de certifier et d'assurer qu'il possède les droits nécessaires pour demander des certificats. En résumé, cette autorité a pour tâche de gérer les requêtes de certificat qu'elle reçoit des différentes entités et de concevoir les paires de clés qui leur sont spécifiques.

2.2.3 – Les certificats

Ils assurent la sécurité d'une clé publique afin d'éviter les failles de sécurité liées à l'usurpation d'identité et à la modification écrite. Leur rôle dans le fonctionnement des PKI sera abordé plus en profondeur dans la suite du document. [20]

2.2.4 – Les services d'archivage et de publication

L'archivage est un service qui permet le stockage des paires de clés pour une restitution en cas de perte de la clé privée. En effet, il a pour mission de stocker en toute sécurité les clés de chiffrement émis au sein de l'infrastructure.

La publication est un service qui répertorie les différents certificats à clés publiques émis par la CA afin de les rendre disponibles aux éventuels futurs utilisateurs, c'est pourquoi on se réfère communément à lui par le terme de dépôt. Ainsi, un annuaire peut être utilisé (LDAP ou X500 par exemple)[21], un serveur Web ou encore un outil de messagerie, etc.

Ce service est contraint par plusieurs exigences telles que, par exemple, le délai de mise à jour des listes de révocation ou la disponibilité de ces listes. Le dépôt est également responsable de la publication de la CRL (Liste de Révocation de Certificat).

2.2.5 – Les utilisateurs

Ce sont les personnes ou entités organisationnelles ayant émis ou émettant des demandes de certificat, ou souhaitant simplement vérifier la validité et les informations sur l'identité d'un certificat préalablement reçu.

En plus des principaux composants que nous venons de voir, nous avons aussi quelques-uns dits complémentaires, à savoir : les bases de données, le serveur d'horodatage, les serveurs HTTP, SMTP, POP.

2.3 – Principe de fonctionnement des infrastructures de gestion de clés publiques (PKI)

Le principe de fonctionnement des infrastructures de gestion de clés repose Essentiellement sur les services précédemment cités.

Les services que l'infrastructure PKI fournit doivent obligatoirement être précédés d'une mise en place d'une entité capable d'effectuer la gestion des différents certificats. Ces services se basent sur des composants tels que nous avons vu préalablement.

Ainsi le fonctionnement d'une PKI se compose de plusieurs étapes :

1. Générer les clés, qui se fait aléatoirement de sorte à garantir leur non-prédictibilité ;
2. Enregistrer les clés, permettant de garder toute leur intégrité et cela de manière confidentielle ;
3. Générer les certificats ;
4. Révoquer un certificat, en cas de corruption de ce dernier. Une fois révoqué celui-ci est consigné dans une CRL (Liste de Révocation de Certificat) ;
5. Supprimer une clé, lorsque celle-ci est expirée ou pose un problème de sécurité.
6. Archiver une clé, afin de garder une trace de celle-ci même après une mise au rencart, afin d'assurer la continuité du travail achevé avec cette dernière.

Nous noterons aussi que de nombreuses applications profitent de la sécurité fournie à travers l'utilisation des infrastructures à clés publiques. Parmi elles, nous avons retenu :

- <u>L'accès à Internet</u>

À travers les navigateurs et serveurs Web qui utilisent le chiffrement pour l'authentification et la confidentialité, mais surtout au niveau du e-commerce qui incite à des transactions financières : ceci implique l'utilisation de protocoles tels que SSL (Secure Sockets Layer), qui permet d'effectuer des échanges sécurisés sur Internet.

- La messagerie

Afin de sécuriser la messagerie, l'utilisation des paires de clés est nécessaire pour la sécurisation des messages, fichiers et signatures. Le protocole utilisé est le S/MIME (Secure Multipurpose Internet Mail Extensions), ce protocole gére la confidentialité des courriels.

- Le réseau privé virtuel

Le chiffrement des données et l'authentification sont les deux principales fonctions utilisées pour gérer les liens entre les différentes parties au sein d'un réseau privé virtuel (Virtual Private Network (VPN)). Afin d'assurer la confidentialité entre les paires ou équipements (site-to-site, router-to-router) et pour sécuriser les connexions à distance (client-To serveur). Cependant, l'IETF a intégré ces services dans le protocole IPSec afin de la sécuriser les tunnels VPN.

Le fonctionnement des infrastructures à clé publique repose fondamentalement sur les mécanismes de gestion des certificats et de signature numérique.

Un certificat numérique est un document électronique permettant l'association entre une clé publique et une entité (personne, équipement (dans le cas du réseau V2G), entreprise…) afin d'assurer sa validité. On peut donc établir de façon triviale que le certificat est le lien entre une entité physique et une entité numérique, certifié par le CA.

Il existe plusieurs types de certificats qui sont présentés en différentes classes selon le niveau de sécurité :

- Classe 1 : Requiert une adresse courriel du demandeur ;
- Classe 2 : Exige une preuve d'identité du demandeur (document D'identification, numéro de série unique, etc.) ;
- Classe 3 : Exige la présence physique du demandeur.

Comme présenté dans la figure 1, les certificats ont un cycle de vie composé des phases suivantes :

1. Demande de certification ;
2. Vérification des attributs ;
3. Création et signature du certificat ;
4. Remise au demandeur (publication) ;
5. Utilisation du certificat ;
6. Suspension ou révocation du certificat ;
7. Expiration du certificat (possible renouvellement).

Figure 10: Cycle de vie d'un certificat

Le standard régissant les certificats numériques est le X.509. Il s'agit d'une norme cryptographique soumise par l'UIT (Union Internationale des Télécommunications) mise en place dans les PKI notamment. La norme X.509 définit un certificat en plusieurs champs :

- Numéro de version : identifie la version du X.509 ;
- Numéro de série du certificat : propre au CA ;
- Identifiant de l'algorithme du certificat : algorithme utilisé pour la signature ;
- Émetteur : « Distinguished Name » (DN) du CA émettant le certificat ;
- Période de validité : intervalle de temps représentant la durée de vie du certificat ;
- Demandeur : DN du détenteur de la clé publique ;
- Information de la clé publique du demandeur : nom de l'algorithme à clé publique, paramètres concernant la clé.

Les certificats émis le sont pour une durée déterminée et suivent le cycle de vie abordé précédemment. Les certificats peuvent être suspendus à la demande du détenteur ou pour des raisons de sécurité.

La signature numérique est définie comme « un mécanisme permettant de garantir l'intégrité d'un document et d'en authentifier l'auteur ». Ainsi, une signature électronique ne peut être falsifiée, réutilisée dans un autre document, reniée, ni altérée.

Lorsque l'on parle de signer numériquement un document, il s'agit alors d'effectuer un chiffrement du dit document à l'aide de sa clé privée, qui est connue uniquement du propriétaire légitime. La signature est infalsifiable, car c'est la clé privée qui l'a générée au moment de la signature. La signature assure, par ailleurs, l'intégrité du document du fait que toute altération serait automatiquement décelée lors du déchiffrement. Il est tout de même préférable d'effectuer la signature sur le hash du message, obtenu à travers une fonction de hachage, à envoyer afin de réduire le temps d'exécution de l'opération tout en garantissant l'authenticité de celui-ci, étant donné les chances infinitésimales d'obtenir un même hash à partir de deux messages différents. Ainsi à la réception du document, il est possible d'identifier l'entité émettrice à l'aide de sa clé publique, et de s'assurer de la non-altération du message grâce à l'empreinte de celui-ci.

Une fonction de hachage aussi appelée fonction de condensation est une fonction permettant d'obtenir un condensat (ou empreinte) d'un texte, c'est-à-dire une suite de caractères relativement courte spécifique au message (ou document) condensé. Il s'agit d'une fonction mathématique à sens unique qui permet de calculer à partir d'une suite binaire une chaîne de caractères qui fera office d'empreinte numérique. Ainsi lorsque la suite binaire est altérée, l'empreinte l'est aussi. Les fonctions de hash les plus utilisées sont :

- MD5 (Message Digest), créant une empreinte d'une taille de 128 bits
- SHA (Secure Hash Algorithme), créant des empreintes à partir de 160 bits

2.4 – Les modèles et les architectures PKI

Les relations entre les composants de l'infrastructure à clé publique sont catégorisées en modèle et architecture selon la situation dans laquelle l'infrastructure PKI est mise en place.

Chaque autorité de certification à un nombre d'entités avec lesquelles elle communique, qui permet le contrôle plus ou moins aisé des échanges.

2.4.1 – Les modèles

Les modèles les plus couramment utilisés sont appelés « Trust Models » ou modèles de confiance, car ils ont été maintes fois éprouvés. Les « Trust Models » sont les suivants :

- Certificate Trust List (CTL)

  Le modèle de confiance basé sur la liste des certificats de confiance demande dans un premier temps que l'utilisateur envoie sa clé publique à l'autorité de certification (CA) via un canal sécurisé. Après réception, l'autorité de certification envoie à l'utilisateur sa clé publique en plus d'un certificat qu'elle a signé avec sa clé privée. L'autorité de certification envoie ensuite sa clé publique au gestionnaire de liste de confiance (GLC), puis le GLC envoie sa propre clé au système de vérification et tout cela se fait sur un canal de transmission sécurisé.

Entre temps, le GLC utilise une fonction de hachage pour créer une empreinte de la clé publique du CA et l'envoie au système de vérification qui dès lors dispose de tous les éléments pour vérifier l'identité des interlocuteurs (CA, GLC, utilisateur). Aussi, l'utilisateur peut effectuer tout envoi de données et il suffira alors qu'il signe et joigne le certificat du CA à son message.

- Certificate Request Message

Le fonctionnement de ce modèle est presque similaire au précédent : en premier lieu, l'utilisateur envoie sa clé publique vers une autorité, cette fois-ci, l'autorité d'enregistrement des certificats (RA), mais toujours à travers une liaison sécurisée.

La RA transfère un message signé au CA ; ce message inclut, en plus des informations concernant l'utilisateur, la clé publique de l'utilisateur : c'est ainsi qu'est émise la demande de certificat. L'utilisateur recevra alors de la part de l'autorité de certification, un certificat signé.

Enfin, après que le CA ait envoyé sa clé publique au système de vérification, l'utilisateur pourra transmettre les données qu'il devra signer et accompagner du certificat qui lui a été délivré afin que le système de vérification puisse confirmer leur authenticité.

- Out of Band Mechanism (OOB)

C'est un modèle de confiance qui permet de créer une empreinte de la clé d'une CA de manière sécurisée (à l'aide d'une fonction de hachage). Une fois créée, la clé peut être acheminée sur un réseau peu, voir non sécurisé.

Le destinataire à qui on a au préalable fourni les informations concernant la fonction avec laquelle cette empreinte a été générée pourra comparer à la réception et vérifier si les données n'ont pas été altérées afin de pouvoir en récupérer de manière sûre la clé de la CA.

Cette technique est communément utilisée pour sécuriser les protocoles sur les navigateurs et serveurs Web.

- Cross Certification

Le modèle de certification croisée propose une architecture à deux CA. Elle est plus sécurisée et permet d'avoir les traces de toutes les transactions effectuées.

L'utilisateur envoie sa clé publique à une première autorité de certification qui lui renvoie un certificat signé avec sa clé privée. Via des liaisons sécurisées, la clé publique du CA1 est transmise au CA2 et celle du CA2 est envoyée au système de vérification. Le CA2 établit et envoie au système de vérification un certificat croisé, c'est-à-dire la clé publique de la première autorité signée de la clé privée de la seconde autorité. Enfin, l'utilisateur peut envoyer des données qu'il aura préalablement signées et accompagnées du certificat qui lui a été fourni. Le système de vérification se charge désormais de la sécurité lors de l'échange des données.

2.4.2 – Les architectures

L'infrastructure PKI est généralement composée de plusieurs CA reliés par des « trust paths » ou chemins de confiance. Selon l'environnement, les CAs peuvent être organisés de manière complètement différente et de leur architecture dépendra l'adaptabilité du modèle de confiance. Ainsi, les architectures les plus couramment utilisées sont les suivantes :

- L'architecture hiérarchique

Le fonctionnement de cette architecture dans le cas de deux autorités de certification (CA1 et CA2) régies par une autorité de certification centrale ou « CAroot » est le suivant : CA1 et CA2 envoient leur clé publique au CA central qui génère un certificat pour chacun des deux CA. Au sein de cette architecture, le « CAroot » a le plus haut niveau d'autorité et possède donc un certificat autosigné. Aussi, cela implique que tous les composants de l'architecture placent leur confiance dans le CA central.

- L'architecture P2P (Peer-to-Peer)

En opposition à l'architecture hiérarchique, l'architecture Peer-to-Peer place les différents CA au même niveau d'autorité. On arrive alors à une situation dans laquelle les certificats sont cosignés, le CA1 pouvant signer des certificats pour le CA2 et vice-versa. L'inconvénient de cette architecture est alors le besoin d'échange mutuel des différentes clés publiques pour qu'un CA génère des certificats pour ses homologues.

- L'architecture en pont

L'architecture en pont ou « Bridge » est une association des deux architectures précédemment abordées. Comme l'architecture hiérarchique a pour principales lacunes la disponibilité et la sécurité et que le modèle pair-à-pair est ralentit par la multitude d'échanges qui y sont générés, alors l'architecture en pont palie aux lacunes des deux architectures précédentes.

Son fonctionnement est similaire à celui du P2P à la différence que les échanges entre CA qui ralentissaient le P2P sont réduits dans la mesure où les CAs n'échangent leurs clés qu'avec l'autorité pont. On peut aussi définir cette architecture comme une architecture hiérarchique où le CAroot est au même niveau d'autorité que les autres CAs qui y sont affiliés.

La figure 2 et le tableau 2 présentent une comparaison des trois architectures citées ci-haut.

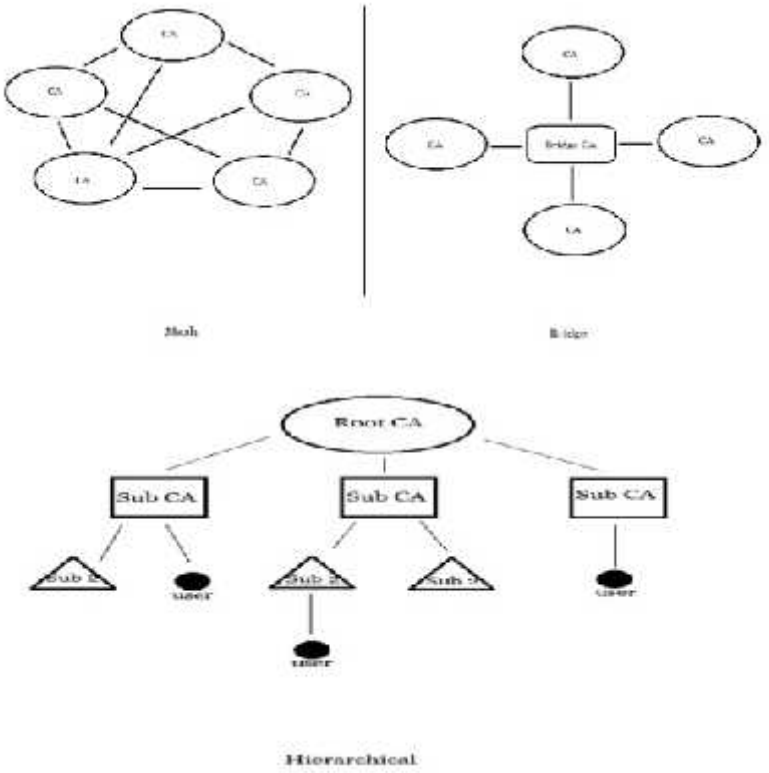

Figure 11: Les architectures PKI [21]

Type d'architecture	Avantages	Inconvénients
Classique (Hiérarchique)	• Adapté à une grande implémentation	• Non flexible • Problème en cas d'indisponibilité du Root CA
Mesh	• Flexibilité	• Non adapté à une grande implémentation • Difficulté à remonter une chaîne de confiance
Bridge	• Interopérabilité entre PKI • Flexibilité • Adapté à une grande implémentation	• Problème en cas • d'indisponibilité du Bridge CA

Tableau 2: Comparaison des architectures PKI

2.5 – Conclusion

Dans ce chapitre, nous avons présenté de manière détaillée les infrastructures à clés publique.

CHAPITRE III : Analyse et conception

1 Introduction

La gestion de projet ou conduite de projet est une démarche visant à structurer, assurer et optimiser le bon déroulement d'un projet suffisamment complexe pour devoir être planifiée dans le temps. L'analyse et la conception sont des parties primordiales dans le cycle de vie d'un projet. Pour la réalisation de notre système de cryptographie hybride, nous avons choisi d'axer notre travail sur la méthode de développement 2TUP qui est née des travaux poussés vers la standardisation et la flexibilité, et ce pour répondre aux contraintes actuelles de gestion et de développement.

2 Définition de 2TUP (Processus de développement)

2TUP signifie « 2 Track Unified Process » .C'est un processus de développement logiciel qui met en œuvre la méthode du processus Unifié.

Le **2TUP** propose un cycle de développement en Y, qui dissocie les aspects techniques des aspects fonctionnels. Il commence par une étude préliminaire qui consiste essentiellement à identifier les acteurs qui vont interagir avec le système à construire, les messages qu'échangent les acteurs et le système. Il s'agit des « chemins fonctionnels » et « architecture technique », qui correspondent aux deux axes de changement imposés ausystème d'information[2] .

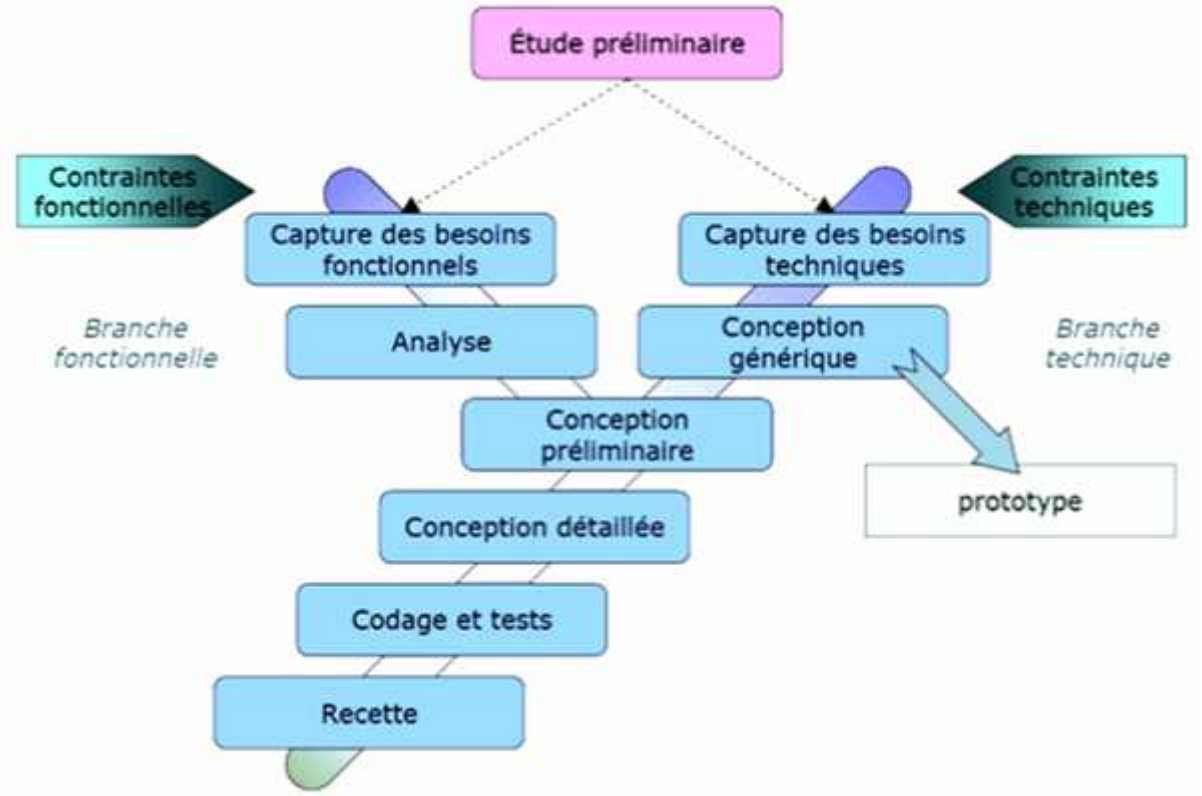

Figure 12:Le processus de développement en Y

3 L’étude préliminaire

3-1 Déterminisation le besoin

Le Centre de Développement des Technologies Avancées CDTA, Nécessite un système de protection de haut niveau ils ont voulu la développer, en développant un nouveau système cryptographique appelé 'hybride', qui combine la rapidité de la cryptographie symétrique avec la sécurité fournie par la cryptographie asymétrique. L’échange de clé se fera via le serveur PKI et cette solution doit être intégrée dans la plateforme de CDTA.

3-2 Clarifier le travail à effectuer

Le projet fin d’étude de Master vise la proposition d’une solution qui permet aux responsables d'assurer l'intégrité et la confidentialité sur les donnes partagé entre les employeurs. Cela se fait par des mécanismes et des algorithmes de sécurité, en particulier la cryptographie hybride (symétrique et asymétrique).

Pour garantir l’identité de récepteur, la solution doit être basée sur une infrastructure de gestion de clés, que le Centre de Développement des technologies avancées le dispose dans son Data Center. Des configurations devront être misent en place pour délivrer des certificats de type utilisateur.

3-3 Captures des besoins fonctionnels

La capture des besoins fonctionnels est la première étape de la branche gauche du cycle en Y. Elle formalise et détaille ce qui a été ébauché au cours de l'étude préliminaire. Focalisée sur le métier des utilisateurs. Elle qualifie plus tôt le risque d’un système inadapté aux utilisateurs. Notre analyse consiste à étudier précisément la spécification fonctionnelle de manière à obtenir une idée de ce que va réaliser le système de cryptographie hybride en terme de métier.

Le processus métier de la cryptographie hybride des fichiers se résume en deux parties essentielles :

a- L’expéditeur :

Lorsque le fichier est envoyé :

- On génère automatiquement un mot de passe aléatoire
- On chiffre le fichier par le mot de passe généré par un algorithme symétrique AES
- On récupère la clé publique du destinataire à partir de son identité ;
- On chiffre le mot de passe généré par la clé publique de destinataire par un algorithme asymétrique RSA.
- On transmet le fichier crypte, le mot de passe crypte, et les deux algorithmes utilisés.

b- Le destinataire :

- On reçoit le fichier crypte, le mot de passe crypte.
- On utilise sa clé privée pour déchiffrer le mot de passe chiffré par algorithme asymétrique RSA.
- On utilise le mot de passe déchiffré pour déchiffrer le fichier par algorithme symétrique AES.
- Reçoit le fichier déchiffré.

3-4 Captures des besoins techniques

La capture des besoins techniques, qui recense toutes les contraintes et les choix dimensionnant la conception du système, les outils et le matériel sélectionnés ainsi que la prise en compte des contraintes d'intégration avec l'existant (pris requis d'architecture technique).

Les choix techniques adoptés pour le projet sont comme suit :

- La modélisation du système avec UML et l'outil visuel paradigme.
- Le langage Python.
- Serveur :
    - PKI,
    - Linux Ubuntu,
    - Library OpenSSL,
    - Protocole SSH.

- Les API : Django qui définit l'architecture générale du Framework.
- Base de données : PgAdmin4.
- IDE : Visual Studio Code.
- L'algorithme AES pour le cryptage symétrique.
- L'algorithme RSA pour le cryptage asymétrique.

3-5 Architecture serveur PKI

La figure ci-dessous présente l'architecture de serveur PKI, est une infrastructure à clé publique se présente en un ensemble de services externalisés qui assurent une meilleure gestion des principaux critères sur la sécurité des réseaux tels que l'authentification ou encore l'intégrité.

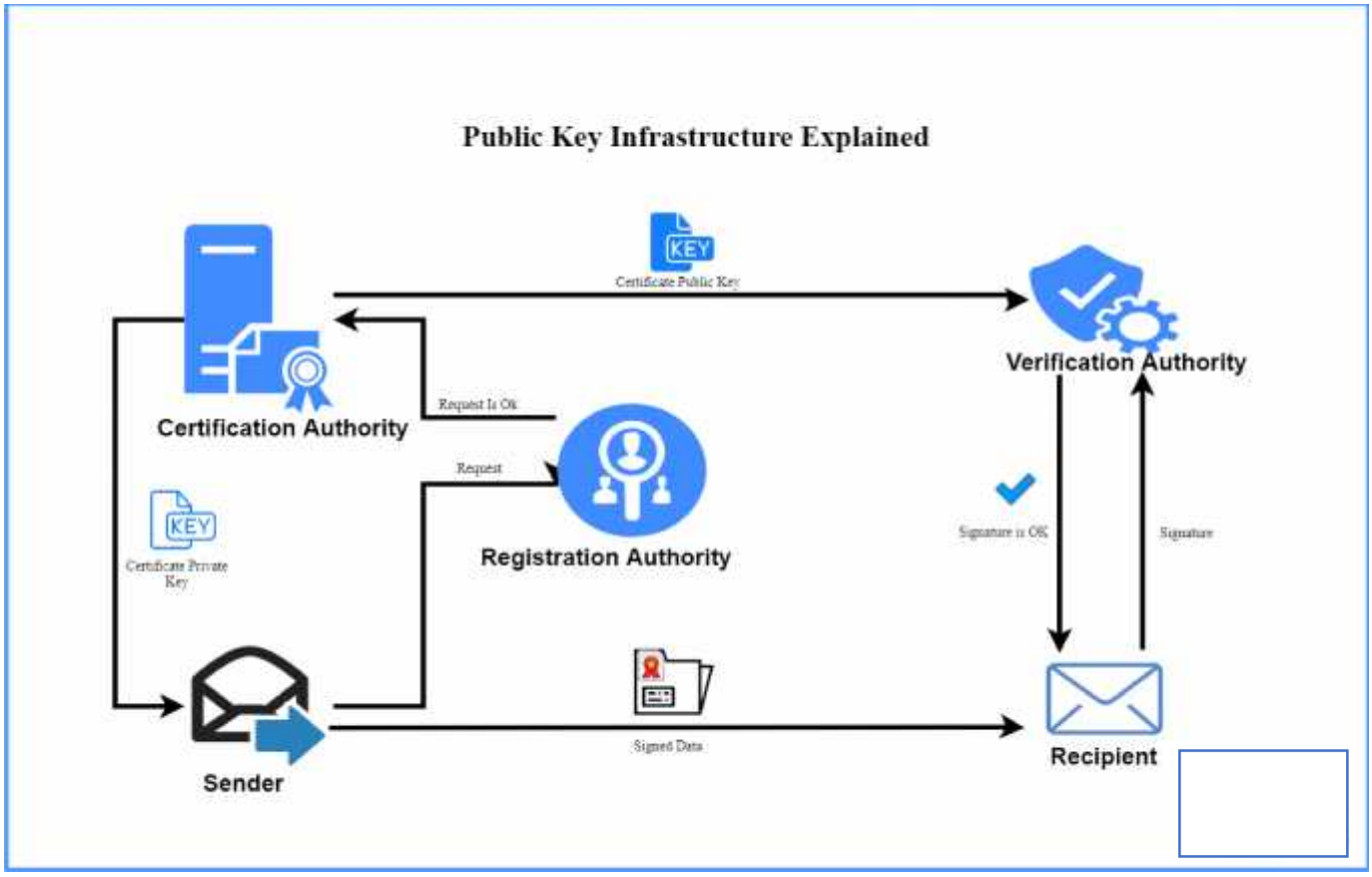

Figure 13 : Architecture serveur PKI

3-6 Analyse fonctionnelle et définition des objectifs

La capture des besoins fonctionnels, produit un modèle de besoins focalisé sur le métier des utilisateurs. Cette étape élimine le risque d'avoir un système inadapté aux besoins des utilisateurs.

3.6.1Identification des cas d'utilisation

3.6.1.1 L'outil Visual Paradigme

Pour une bonne présentation des diagrammes UML on a choisi Visual paradigme version 16.3 qui est un logiciel de création de diagrammes, conçu pour les développeurs logiciels afin de modéliser des systèmes et de gérer les processus de développement. Le logiciel présente de nombreux repères pour accéder facilement à ses fonctionnalités.

3.6.1.2 Diagramme de cas d'utilisation

Le modèle de cas d'utilisation capture les exigences d'un système et les fonctionnalités futures que doit l'implémenter. Les cas d'utilisations sont un moyen de communication avec les utilisateurs et d'autres parties prenantes ce que le système est destiné à faire. Ils montrent l'interaction entre le système et les entités externes au système. Il scinde la fonctionnalité du système en unités cohérentes.

3.6.1.3Identification des acteurs

Le système de cryptographie hybride nécessite trois types d'acteurs principaux dont chacun a son rôle. Le tableau ci-dessus résume le rôle de chaque acteur :

Acteur	Rôle
Administrateur	Il gère les utilisateurs et leurs certificats
Expéditeur	Il choisit le destinataire Il nommé l'objet de fichier Il sélectionner le fichier et envoyer
Destinataire	Il téléchargé sa clé privée consulter le fichier chiffre et le mot de passe chiffre Téléverser sa clé privée et télécharger le fichier

Tableau 3: les acteurs et leurs rôles

3.6.1.4Cas d'utilisation de l'administration des utilisateurs et des certificats

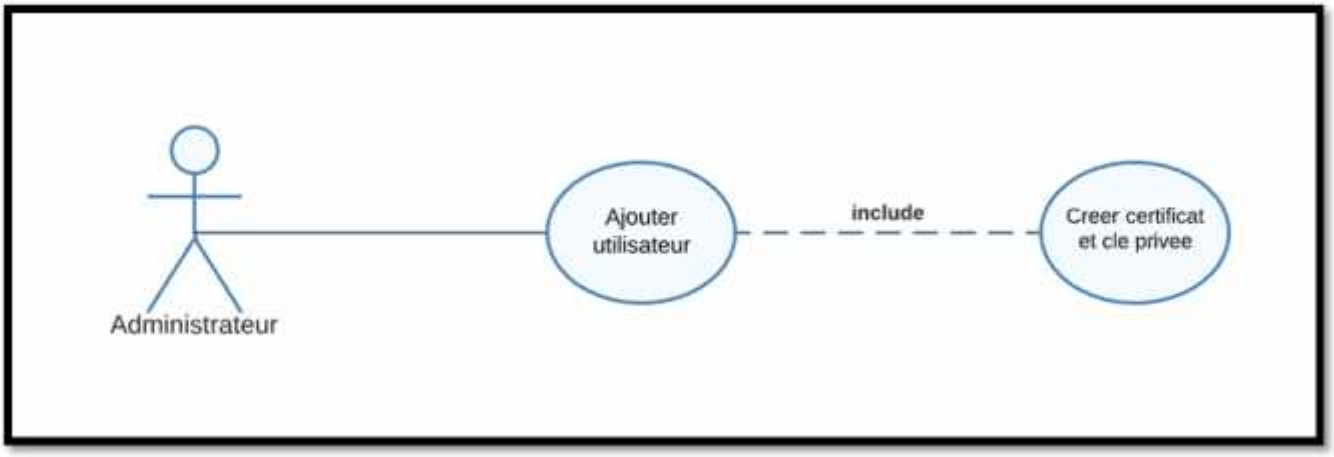

Figure 14. Cas d'utilisation de l'administrateur.

Description des cas d'utilisations de l'administration

Acteur	**Cas d'utilisation**	**Rôle**
Administrateur	Ajouter utilisateur	Saisir les informations etajouter un compte utilisateur. Envoi de certificat et clé privée automatiquement aux utilisateurs.

Tableau 4. Description des cas d'utilisation de l'administration.

3.6.1.5 Cas d'utilisation de l'expéditeur

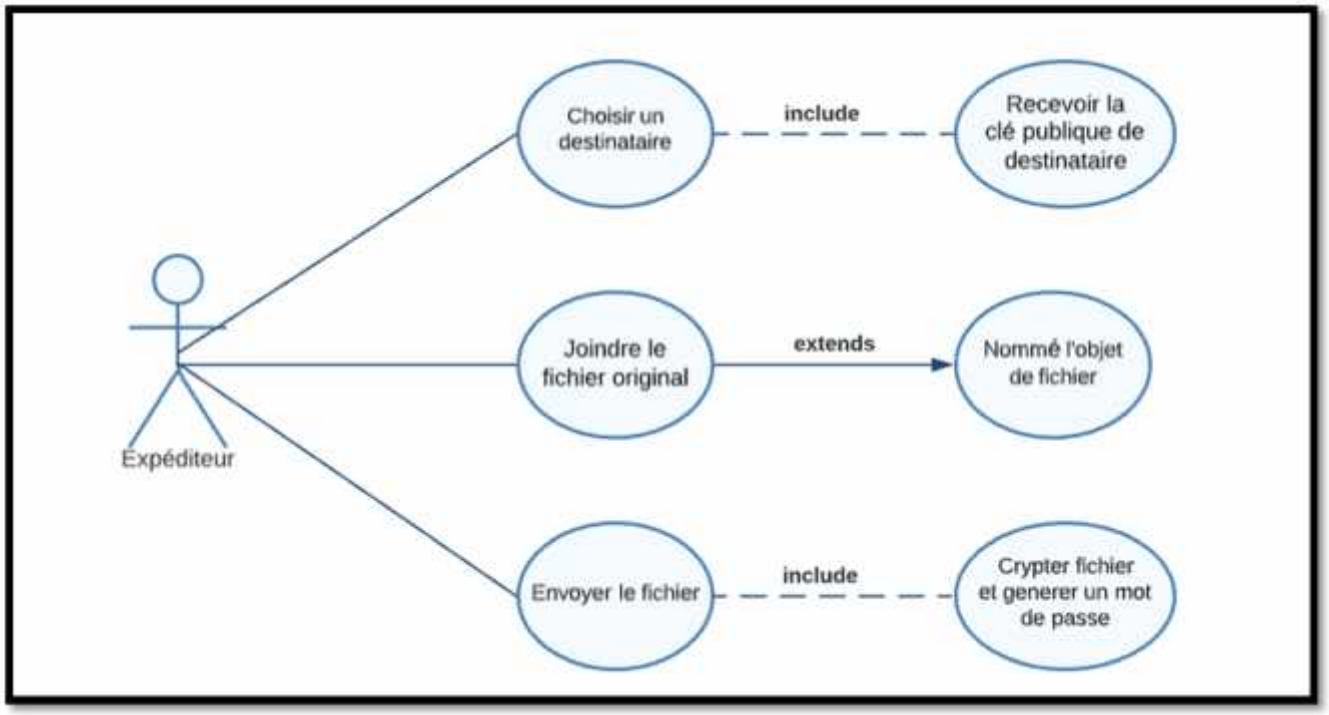

Figure 15 : Cas d'utilisation de l'expéditeur

Description de cas d'utilisation de la expéditeur

Acteur	Cas d'utilisation	Rôle
Expéditeur	Il choisit l'employeur auquel il veut envoyer un fichier	choisi l'utilisateur depuis une liste des employeurs. Reçoit la clé publique de destinataire depuis l'autorité de certification.
	Joindre le fichier original	joindre et nommer le document. Recevoir un mot de passe générer automatiquement aléatoire.

Tableau 5 : Description de cas d'utilisation de l'expéditeur.

3.1.6.6 Cas d'utilisation de destinataire

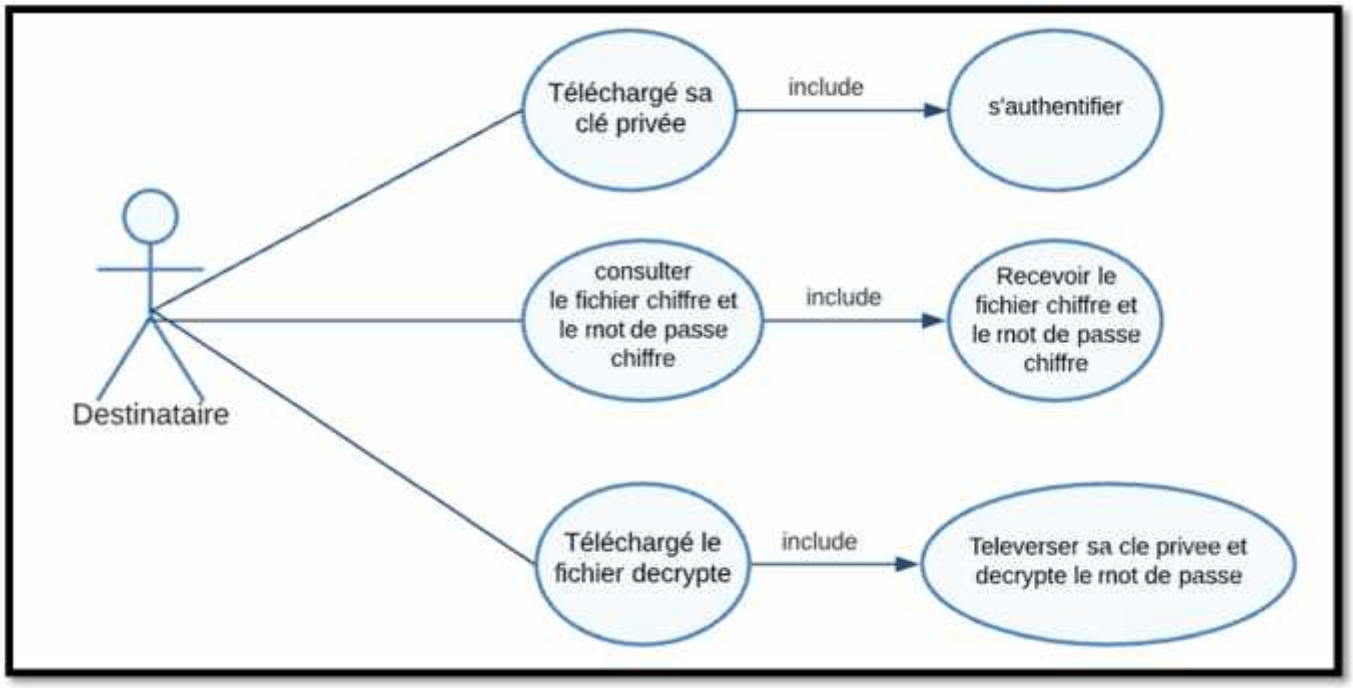

Figure 16 : Cas d'utilisation de destinataire

Description des cas d'utilisations de destinataire

Acteur	Cas d'utilisation	Rôle
Destinataire	Téléchargé sa clé privée	Apres authentification il reçoit sa clé privée depuis l'admin.
	consulter le fichier chiffre et le mot de passe chiffre	Avant le décrypte le fichier il reçoit le fichier chiffre et le mot de passe chiffre.
	Téléverser sa clé privée et décrypte le fichier	Avec sa clé privée décrypte le mot de passe avec algorithme asymétrique RSA. Apres automatiquement le fichier déchiffré avec le mot de passe avec algorithme symétrique AES.

Tableau 6 : Description des cas d'utilisations de destinataire

3.6 L'analyse des besoins

3-7-1 Diagramme d'activité

3-7-1-1 Diagramme d'activité de l'envoie de fichier

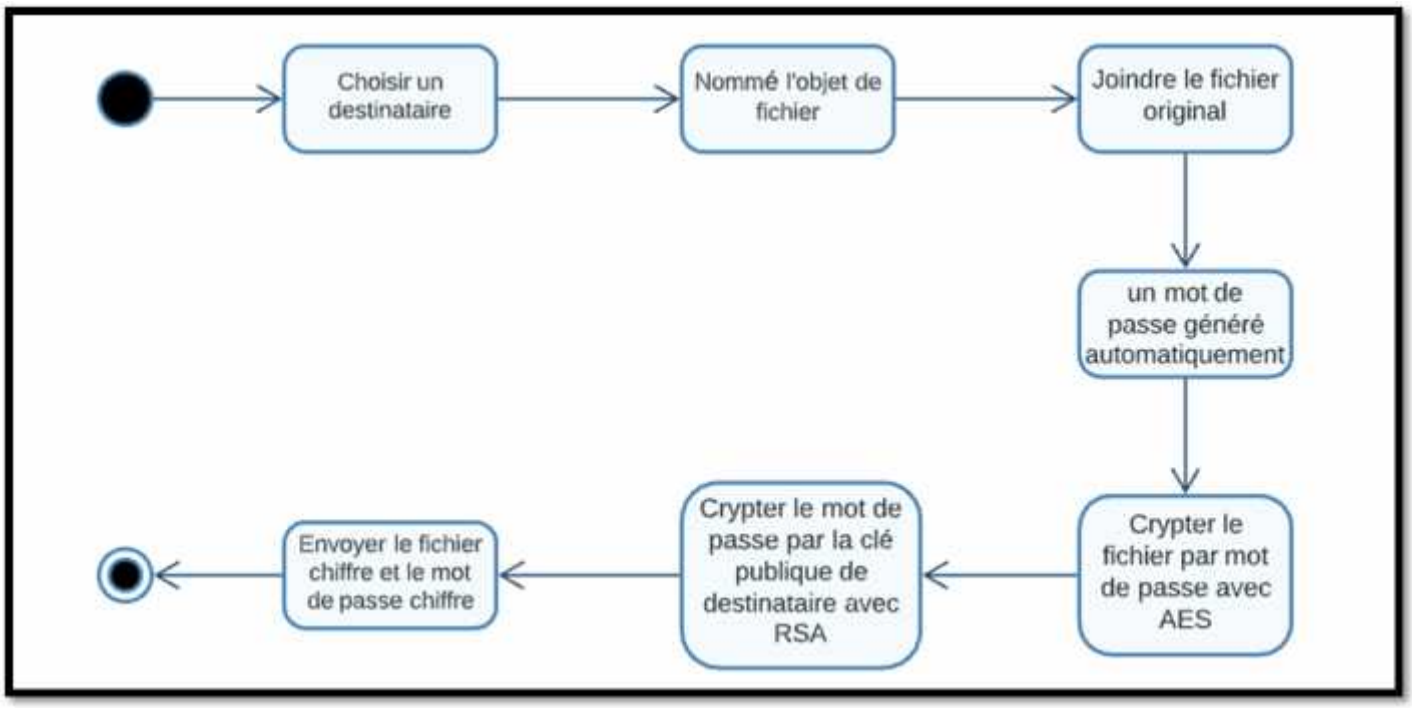

Figure 17 : Diagramme d'activité de l'envoie de fichier

3.6.1.5 Diagramme d'activité de recevoir le fichier

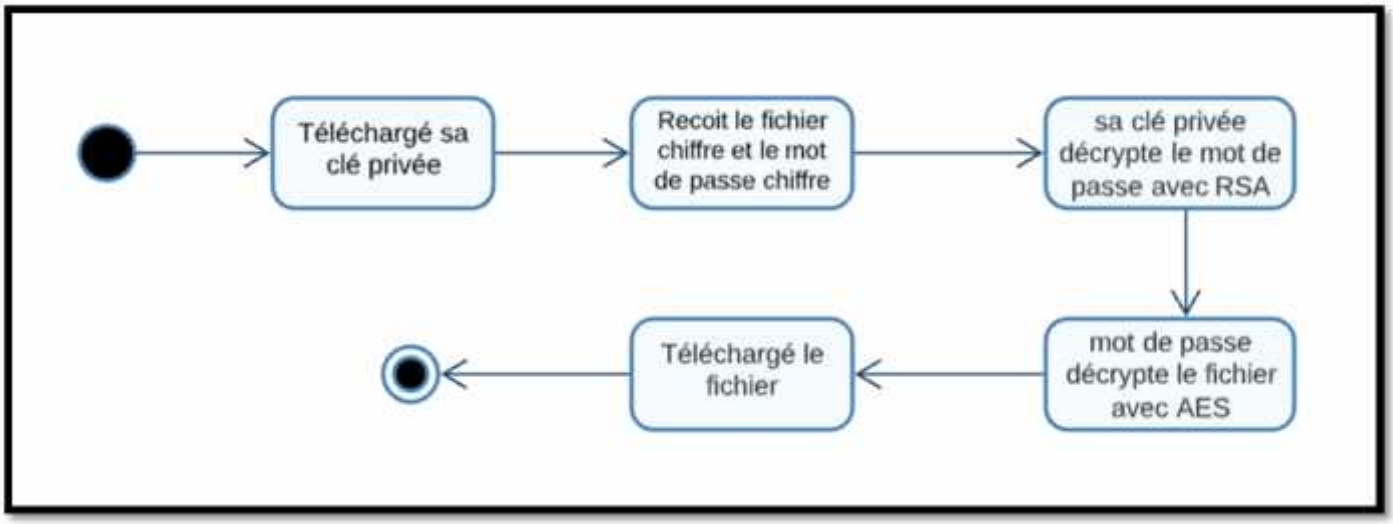

Figure 18 : Diagramme d'activité de recevoir le fichier

3-7-2 diagramme de séquence

Un diagramme de séquence montre chronologiquement (de haut en bas) les interactions entre un ensemble d'objets. Dans cette partie nous décrivons deux diagrammes essentiels (métier) de notre système qui sont, le diagramme de séquence de l'envoie d'un fichier et le diagramme de séquence de la réception d'un fichier en cryptage hybride.

3-7-2-1 Diagramme de séquence de l'envoie d'un fichier

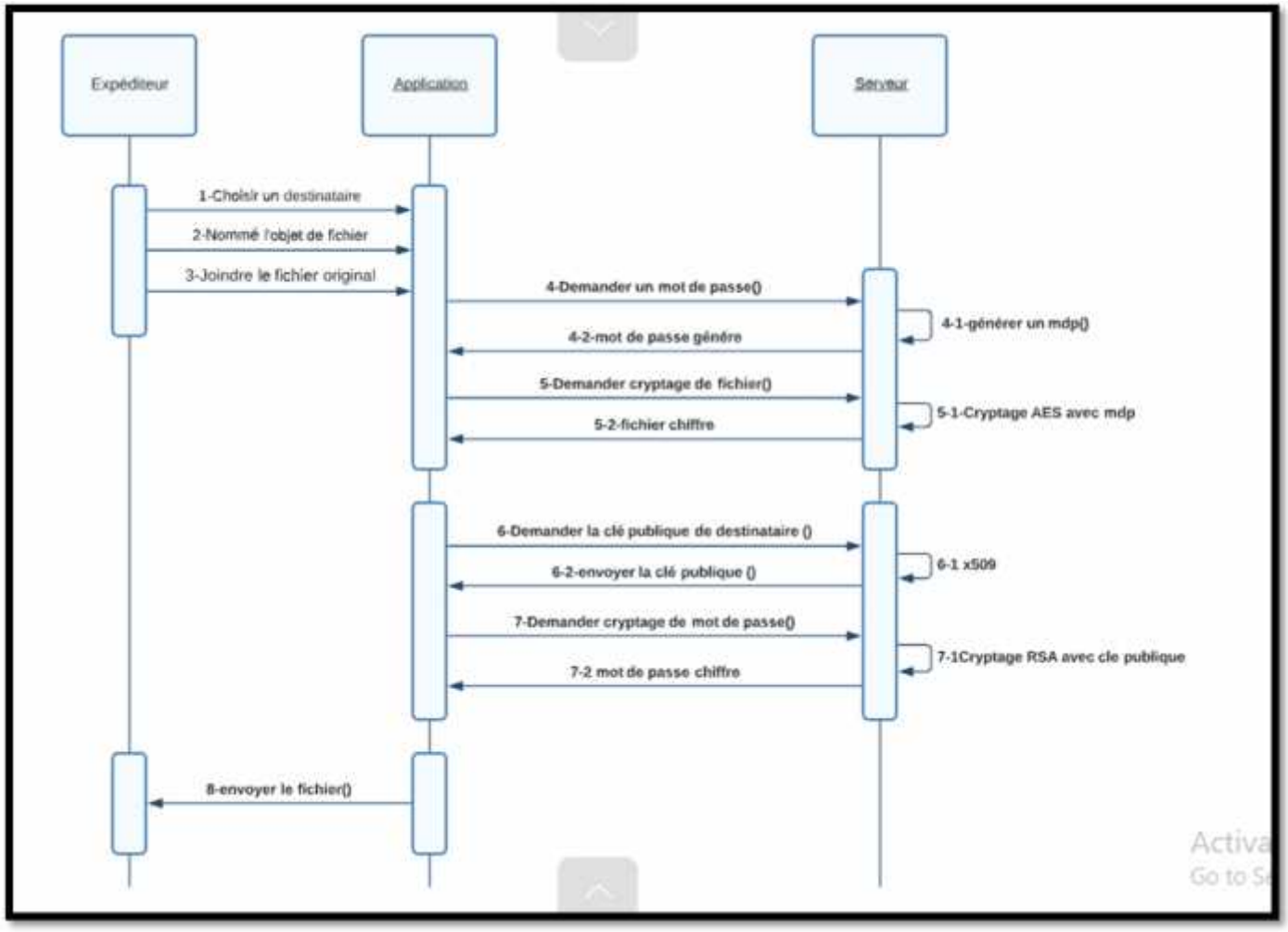

Figure 19 : Diagramme de séquence de l'envoie d'un fichier

3-7-2-2 Diagramme de séquence de la réception d'un fichier

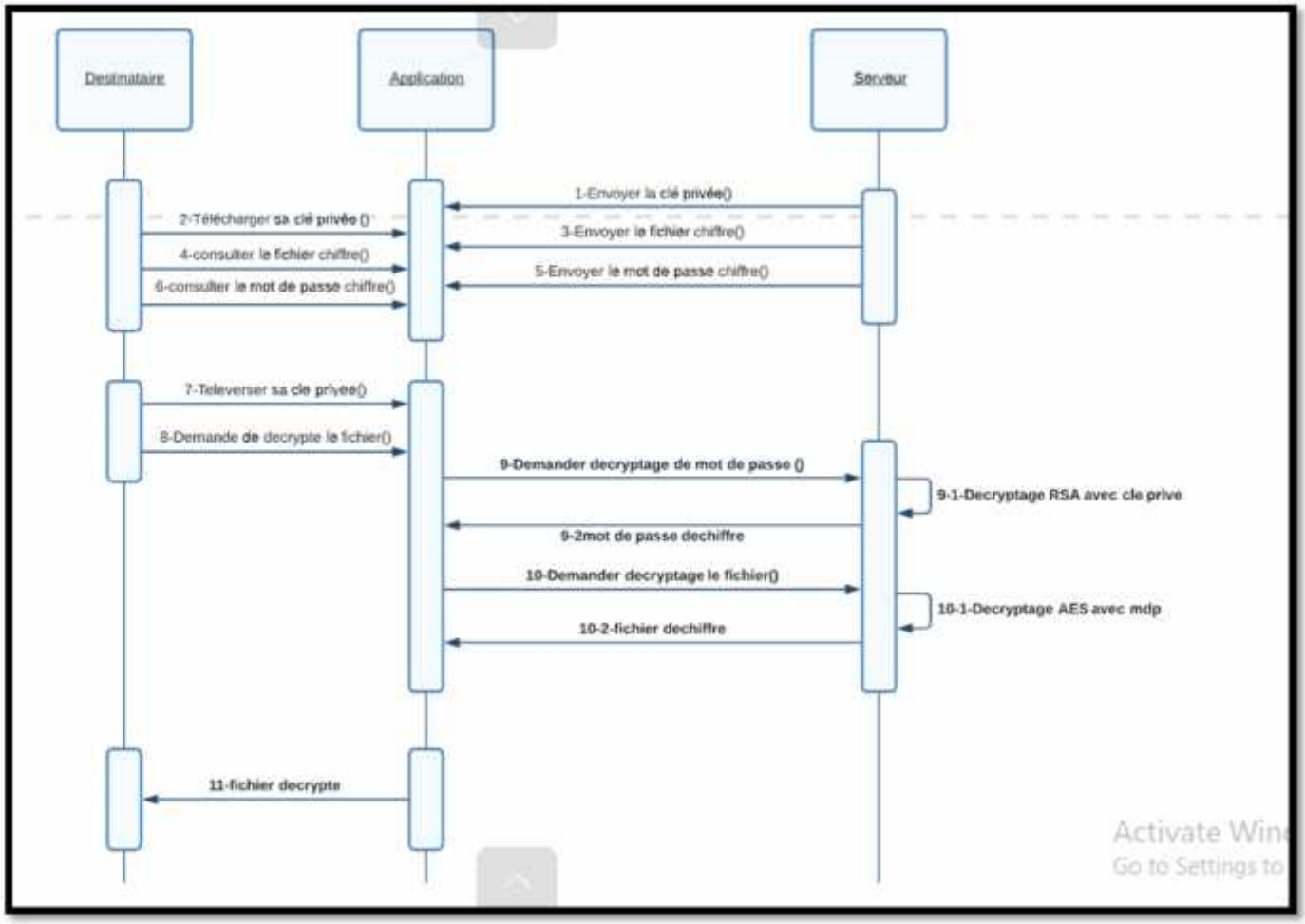

Figure 20 : Diagramme de séquence de la réception d'un fichier

3-8 La Conception

3-8-1 Le Diagramme de classes

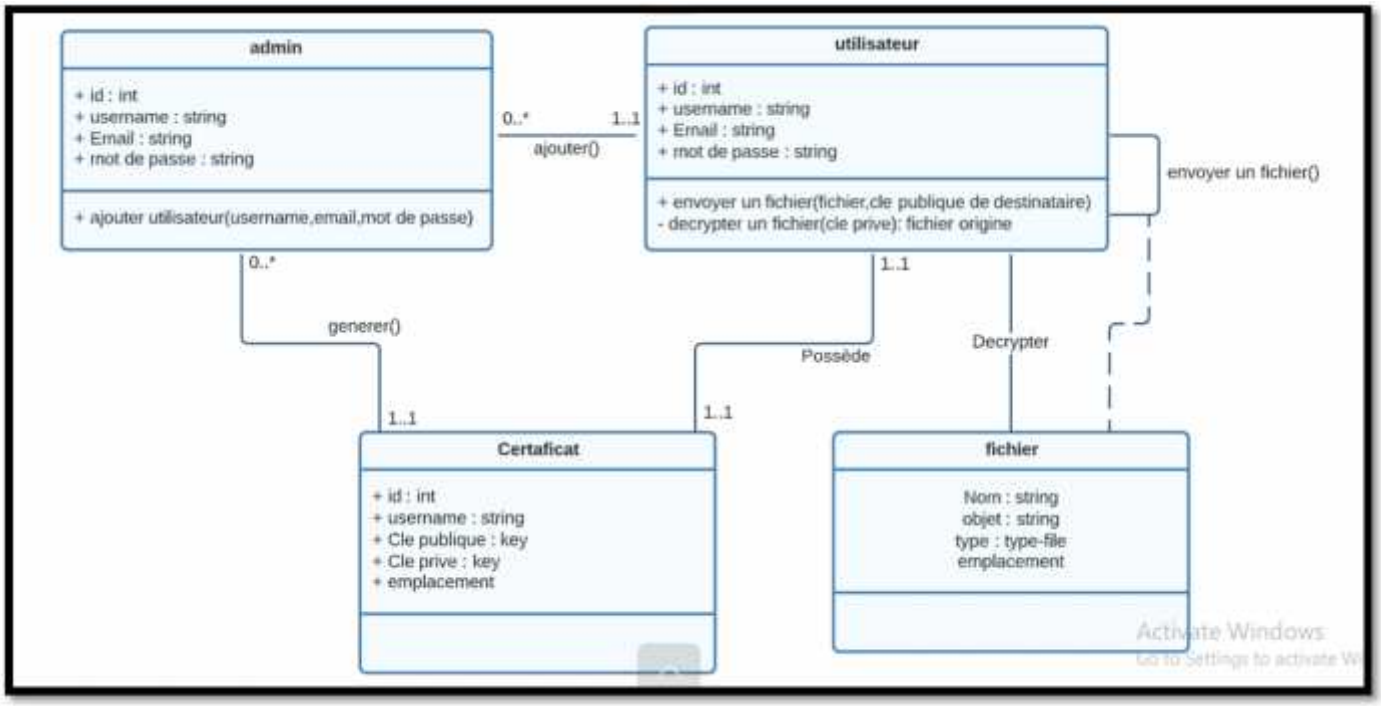

Figure 21 : Diagramme de classes

4 Conclusion

Dans ce chapitre, nous avons modélisé un système de cryptage hybride pour des différents formats de document (Pdf, Doc, Png, Txt), pour se faire nous avons utilisé le langage UML pour la conception de la solution proposée. Dans le chapitre qui suit nous allons aborder la partie réalisation.

CHAPITRE IV : Réalisation

1- Introduction

Le choix du langage s'est porté sur le Python, qui étant un langage de programmation open source à la base et conçu pour être de type sécurisé et facile à utiliser. Il contient une architecture de sécurité destinée à protéger les ressources locales de code chargé à distance. Notre choix s'est porté sur Visual Studio Code, qui nous fournit le confort et la simplicité nécessaires à un développement propre et rapide. Dans le présent chapitre nous présentons le système que nous avons conçu. Les principales interfaces du système sont montrées par des captures d'écran.

2 Les outils utilisés pour le codage et la Base de donnée

2.1 L'éditeur de code Visual Studio Code

Visual Studio Code est un éditeur de code source et environnement de développement qui peut être utilisé avec une variété de langages de programmation, notamment Python, Java, JavaScript, Go, HTML, Node.js et C++. Il est basé sur le cadre Electron, qui est utilisé pour développer des applications Web Django.

Figure 22 : Visual Studio Code

2-2 Django cadre de développement (Framework)

Django est un cadre de développement web *open source* en Python. Il a pour but de rendre le développement web 2.0 simple et rapide. composé de trois parties distinctes :

1. Un langage de gabarits flexible qui permet de générer du HTML, XML ou tout autre format texte ;
2. Un contrôleur fourni sous la forme d'un « *Rempang* » d'URL à base d'expressions rationnelles ;
3. Une API d'accès aux données est automatiquement générée par le cadre compatible CRUD. Inutile d'écrire des requêtes SQL associées à des formulaires, elles sont générées automatiquement par l'ORM.

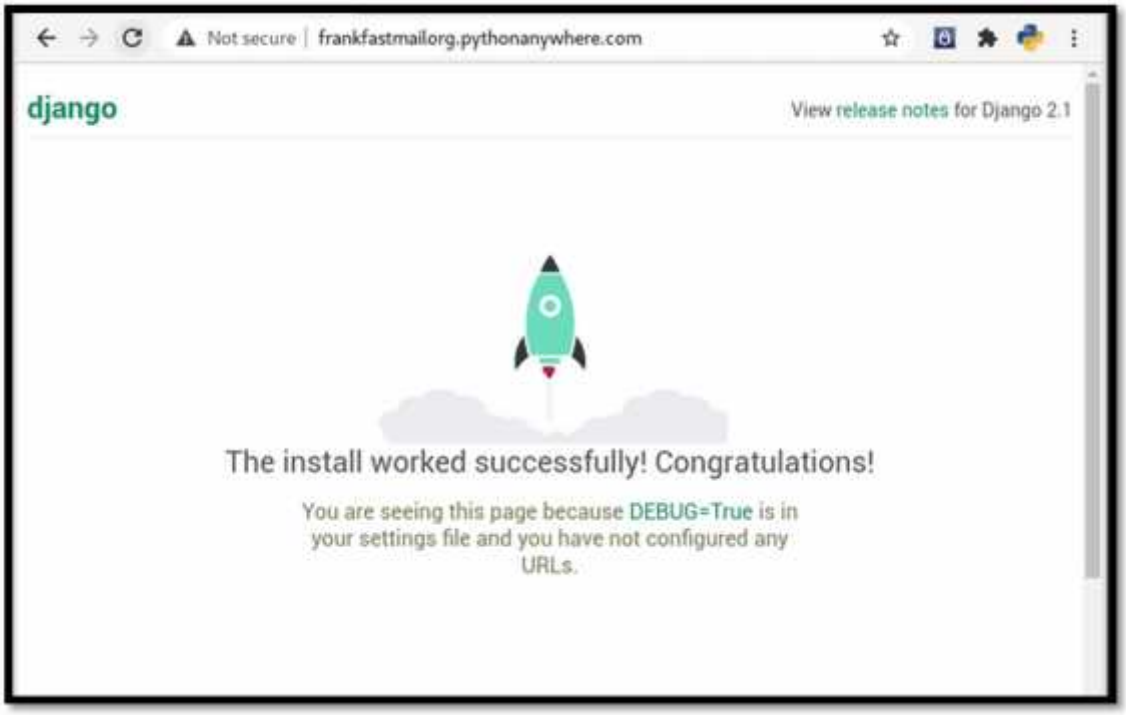

Figure 23 : Django home page

2-3 La bibliothèque pycrypto

Pycrypto est une bibliothèque qui fournit des outils de cryptographie primitive implémententés en langage Python. Il est recommandé d'utiliser une bibliothèque de haut niveau ce qui donnera moins de travail au programmeur pour éviter des longues analyses de sécurité. Pycrypto correspond bien à cette demande mais est une approche dangereuse qui demande de porter un grand détail à son fonctionnement. Cependant il permet de faire ce que l'on désire en cryptographie.

```
hemnada@hemnada-VirtualBox: $ pip install pycrypto
/usr/lib/python3/dist-packages/secretstorage/dhcrypto.py:16: CryptographyDeprecat
tionWarning: int_from_bytes is deprecated, use int.from_bytes instead
  from cryptography.utils import int_from_bytes
/usr/lib/python3/dist-packages/secretstorage/util.py:25: CryptographyDeprecation
Warning: int_from_bytes is deprecated, use int.from_bytes instead
  from cryptography.utils import int_from_bytes
Requirement already satisfied: pycrypto in /usr/lib/python3/dist-packages (2.6.1
)
```

Figure 24 : commande installe pycrypto

2-4 Postgres PgAdmin 4 serveur de bases de données

Postgres **pgAdmin 4** permet de créer toute sorte d'objets du serveur de bases de données PostgreSQL. Ces objets peuvent être des bases de données (BDD), des schémas, des tables, des utilisateurs.. Cet outil permet également d'exécuter des requêtes SQL.

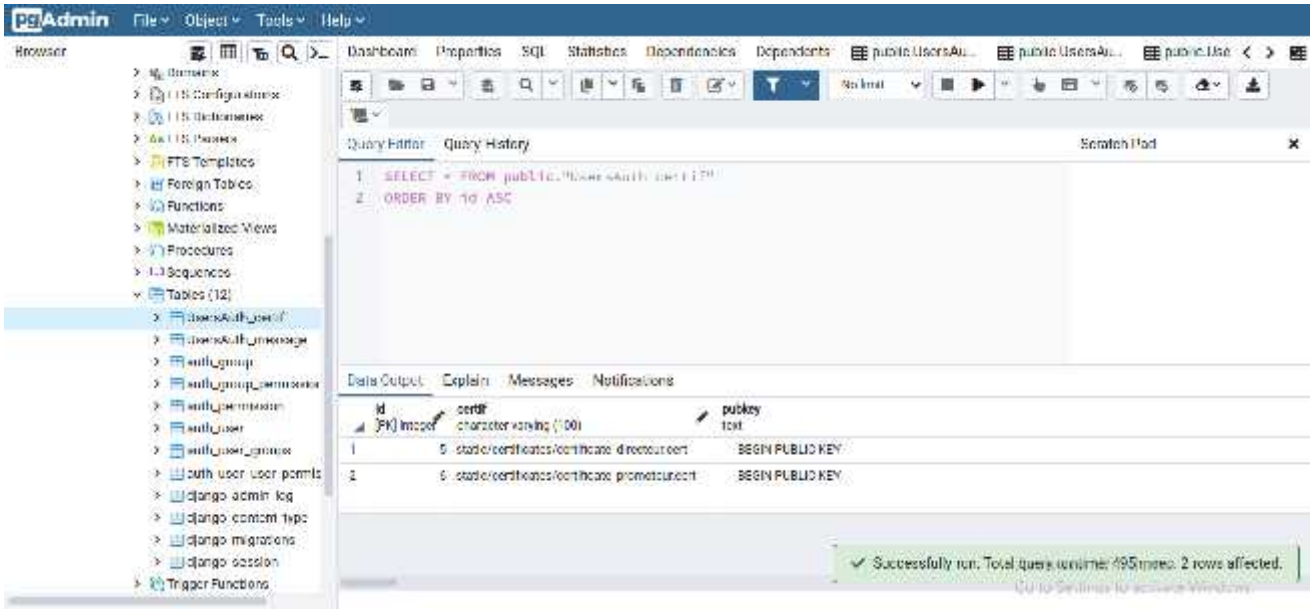

Figure 25 : Postgres pgAdmin 4

3 Les interfaces essentielles de l'application développée

3.1l'authentification

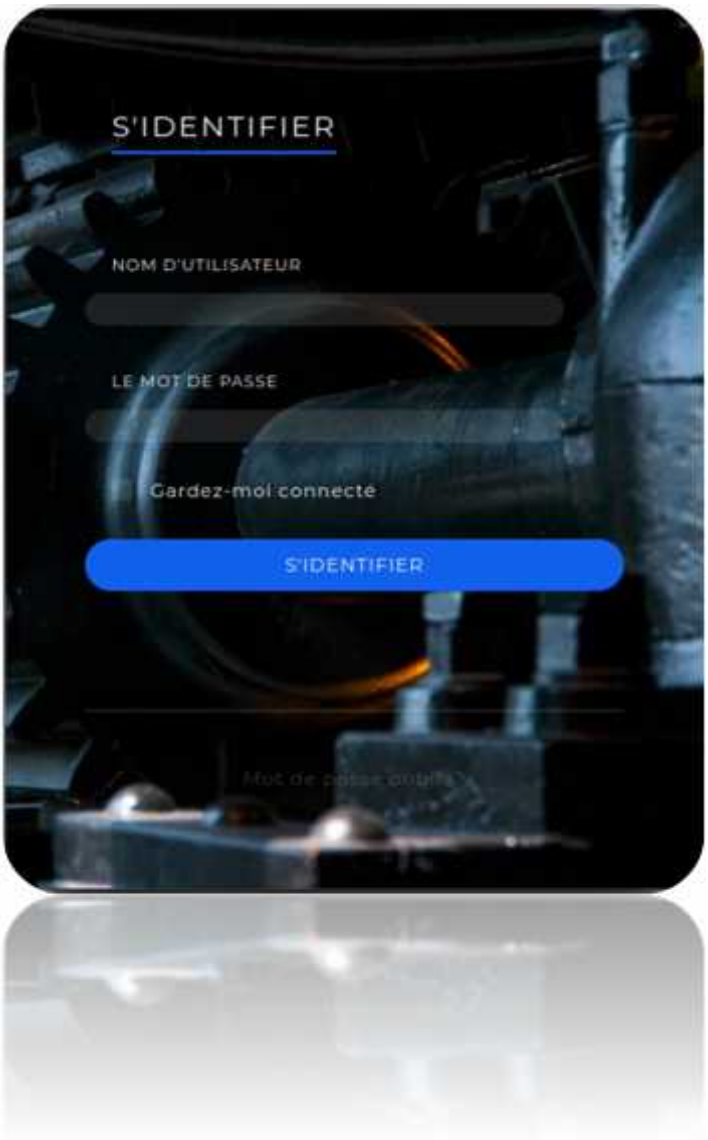

Figure 26 : interface d'authentification

3-2 L'Administration

3-2-1 ajouter un utilisateur

Figure 27 : interface administration

3-2-2 certificat d'utilisateur ajouté généré automatiquement

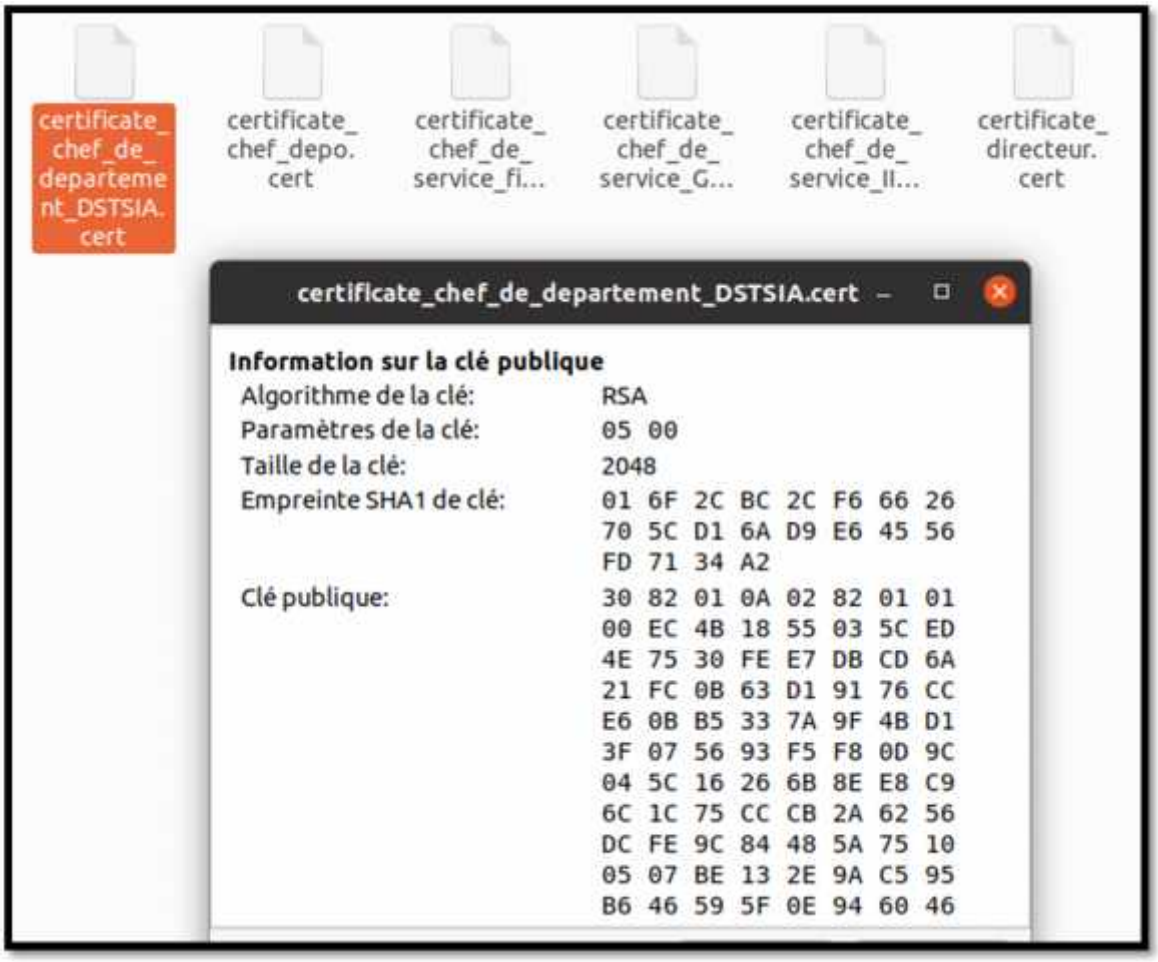

Figure 28 : certificat générer

3-3 premier accès de l'utilisateur à l'application

3-3-1 télécharger la clé privée

Figure 29 : l'interface de premier accès

3-3-2 clé privé téléchargé

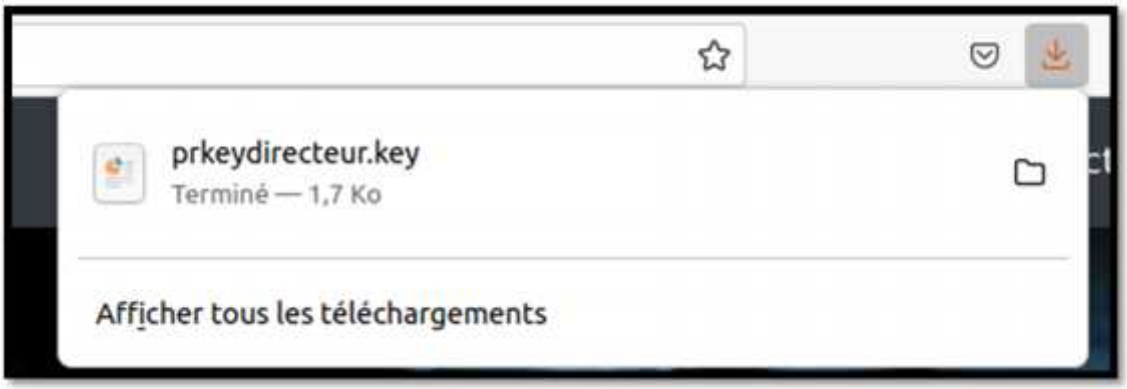

Figure 30 : clé privé téléchargé

3-4 Envoyer un fichier

Figure 31 : interface d'envoie des fichiers

3-5 Les fichiers reçus

Figure 32 : interface de réception des fichiers

3-5-1 le contenue de fichier crypté

Figure 33 : Contenue de fichier chiffré

3-5-2 Le code pour générer un mot de passe aléatoire

```
from OpenSSL import crypto
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad, unpad
from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP
import binascii
from base64 import b64decode, b64encode
import base64

import getpass

try:
    from .models import folder
except:
    pass
import os

caractere=['q','w','c','r','t','y','u','i','o','p','[',']','a','s','d','f','g','h','j','k',

def mdp(longueur):
    mdp=str()
    shuffle(caractere)
    for x in range(longueur):
        mdp+=caractere[x]
    return mdp.encode('utf-8')
```


Figure 34 : Le code pour générer un mot de passe aléatoire

3-5-3 Le code de chiffrement symétrique de fichier AES

```
def encrypt (file_name,key):

    data = file_name
    cipher = AES.new(key,AES.MODE_CFB)
    ciphertext = cipher.encrypt(pad(data,AES.block_size))
    iv = b64encode(cipher.iv).decode('UTF-8')
    ciphertext = b64encode(ciphertext).decode('UTF-8')
    to_write = iv + ciphertext
    return to_write.encode('utf-8')
```

Figure 35 : Le code de chiffrement symétrique de fichier

3-5-4 Le code de chiffrement asymétrique de mot de passe RSA

```
def cypherpass(mp , key):


    pbkey1 = RSA.import_key(key)

    encryptor = PKCS1_OAEP.new(pbkey1)
    if type(mp) == str:
        mp = mp.encode("utf-8")

    encrypted = encryptor.encrypt(mp)#.encode("utf-8"))

    return binascii.hexlify(encrypted).decode('UTF-8')
```

Figure 36 : Le code de chiffrement asymétrique de mot de passe

3.6 décryptage de fichier reçu

3-6-1 téléverser clé prive

Figure 37 : interface de téléverser clé privé

3-6-2 décryptage asymétrique de mot passe RSA

```
def decypherpass(mp , key):
    try:
        data = binascii.unhexlify(mp)
        pvkey1 = RSA.import_key(key)
        decryptor = PKCS1_OAEP.new(pvkey1)

        decrypted = decryptor.decrypt(data)
    except:
        print("wrong Key")
        return "Wrong Key"


    return decrypted
```

Figure 38 : Le code de déchiffrement asymétrique de mot de passe

3-6-3 mot de passe déchiffré

Figure 39 : interface affiche le mot de passe déchiffré

3-6-4 déchiffrement symétrique de fichier AES

```
def decrypt(file_name,key):
    with open(file_name,'r') as entry:
        try:
            data = entry.read()
            length = len(data)
            iv = data[:24]
            iv = b64decode(iv)
            ciphertext = data[24:length]
            ciphertext = b64decode(ciphertext)
            cipher = AES.new(key,AES.MODE_CFB,iv)
            decrypted = cipher.decrypt(ciphertext)
            decrypted = unpad(decrypted,AES.block_size)
            # print(decrypted)
            return decrypted
        except:
            print("wrong password")
```

Figure 40 : Le code de déchiffrement symétrique de fichier

3-6-5 l'accès pour télécharger le fichier décrypté

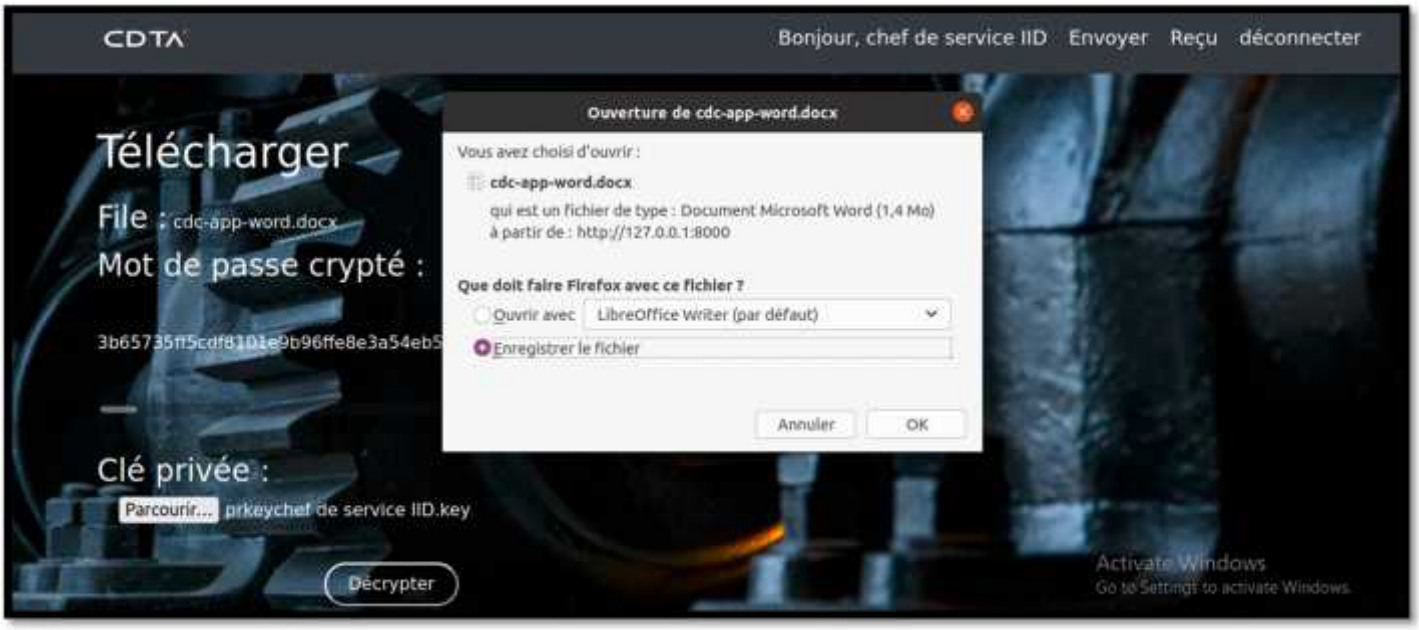

Figure 41 : l'accès pour télécharger le fichier décrypté

3-6-6 le contenu de fichier décrypter

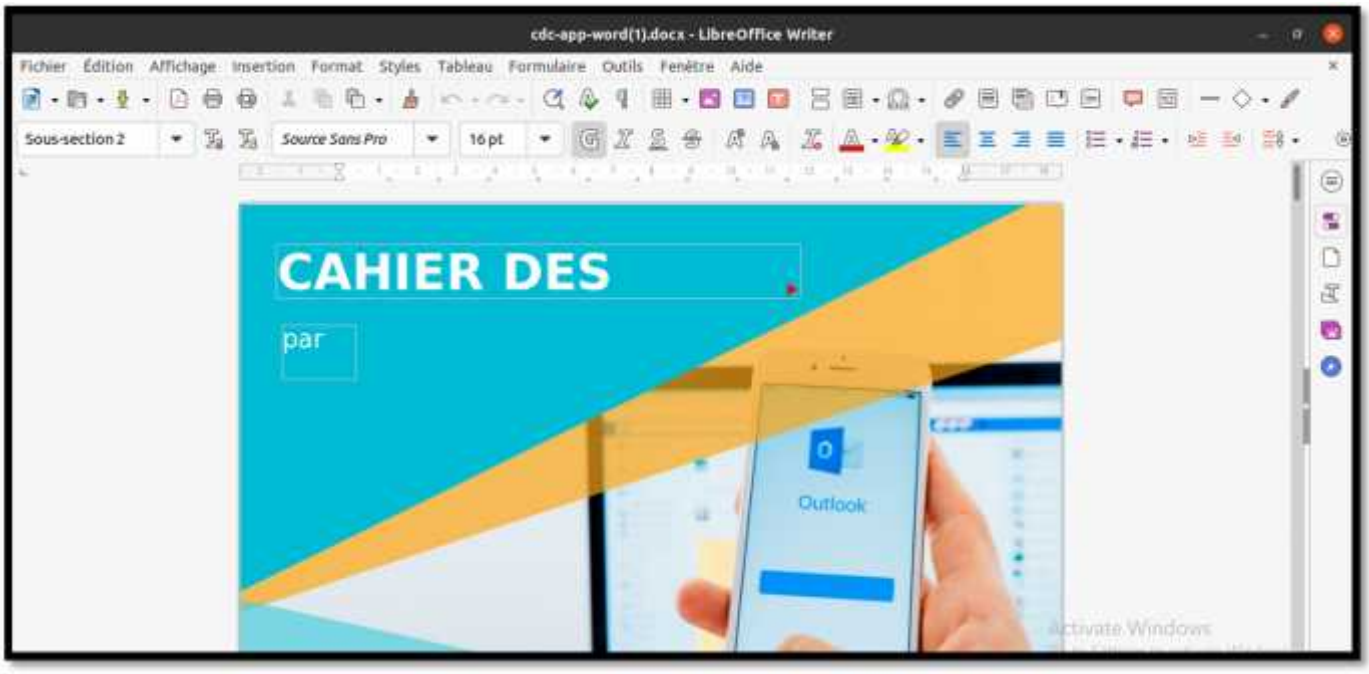

Figure 42 : le contenu de fichier d'origine

4- Conclusion

Dans ce chapitre, nous avons présenté les différents outils et technologies qui ont été utilisés pourréaliser cette solution, nous avons achevé les parties essentielles de la solution en respectant la conception de l'application. Quelques interfaces ont été présentées pour bien clarifier les étapes d'utilisation de la solution proposée dans ce projet de fin d'étude.

Conclusion générale

Notre travail rapporté dans ce mémoire a porté dans une première partie sur l'étude des techniques de cryptage basées sur le serveur de pki et sur les algorithmes de chiffrement AES à clé secrète, l'algorithme RSA à clé publique . Ces deux algorithmes sont nécessaires à la conception de notre cryptosystème..

Dans une seconde partie, nous avons proposé une méthode hybride de chiffrement et de déchiffrement appliquée au chiffrement de fichier et de mot passe génère . Dans cette partie, nous avons élaboré un algorithme hybride associant les algorithmes AES, RSA . Nous avons donné en détail notre cryptosystème , ainsi que son principe de fonctionnement. Puis, nous avons appliqué la technique hybride proposée au chiffrement de fichier . Les résultats obtenus sous django-python ont permis d'illustrer le bon fonctionnement et l'efficacité de la méthode proposée.

Notre travail est considéré comme une première version de projet « cryptographie hybride » au sien de Centre de Développement des Technologies Avancées et au même temps est un maillon très important dans le système de sécurité du Centre.

Ce travail constituant un grand chantier, des améliorations ultérieures sont à venir. Ainsi, en guise de perspectives, nous prévoyons de transformer cette application en plusieurs applets, et on les intègre dans des workfows de différentes algorithme de chiffrement pour la confidentialité des transfert de fichier .

L'amélioration vers la cryptage a triplé algorithme est une étape très importante pour la mise en valeur de la Sécurité de transmission des fichiers .

Les références

1 Bruce Schneier, 'Cryptographie Appliquée, Algorithmes, Protocoles Et Codes Source En C',
Seconde édition, Vuibert Informatique (2001).

2 Toufik Bekkouche, 'Développement Et Implémentation Des Techniques De Cryptage DesDonnées Basées Sur Les Transformées Discrètes', Thèses de doctorat, (2018).

3 Mr Mohamed EL MARRAKI, Nasser Yassine, and Ouyous Mina, 'Rapport Sur L'étude Et L'implémentation De Quelques Algorithmes De Chiffrement Et De Signature',Master informatique,Faculté des science Rabat,2014.

4 Richard A Mollin, *an Introduction to Cryptography*CRC Press, 2000).

5 'Institut D'électronique Et D'informatique Gaspard-Monge (Igm)

6 'Explications Sur La Cryptographie', Tice-Education, (Publication : 16 mai 2020).

7 BESMA DEBBAGH, and NOURA BOUNEGEB, 'Eude Et Comparaison De Principaux Systèmes Crypto Fournis Par Le Package De Bouncy Castle Plateforme Java Sdk',Mémoire MasterII, université Ouargla(2016).

8 YACINE CHALLAL, 'Infrastructures À Clés Publiques', in *Ingénierie des Protocoles et Logiciels Sécurisés*, cours, (2015).

9 Ryma Dr BOUSSAYOUD, Chehla BENHADJI, Labiba CHIOUKH, Doha AFER, Djihan BOUCHAIR, and Nesrine TITI, 'Cryptage/Chiffrement & Tatouage Des Données Numériques', Mémoire MastreII , Université de JIJEL ,(2019).

10 Jérémy Jean, 'Cryptanalyse De Primitives Symétriques Basées Sur Le Chiffrement Aes', *Thesede doct. Ecole Normale Supérieure* (2013).

11 Pierre Barthélemy, Robert Rolland, and Pascal Véron, 'Cryptographie', *Edition Hermès- Lavoisier* (2005). ', (2002).

12 Rodolphe CARDON DE LICHTBUER, 'Signatures Électroniques Dans Les Applications Internet',Mémoire d'ingéniorat, ECOLE ROYALE MILITAIRE Bruxelles, (2006).

13 BESMA DEBBAGH, and NOURA BOUNEGEB, 'Eude Et Comparaison De Principaux Systèmes Crypto Fournis Par Le Package De Bouncy Castle Plateforme Java Sdk', Mémoire Master II, université Ouargla(2016).

14 Saidou Diop, 'Une Infrastructure À Clés Publiques (Pki) Pour Sécuriser Les Messages Dans Un Réseau V2g', Thèse, Université du Québec à Trois-Rivières, (2018).

15 Yevgeniy Dodis, and Nelly Fazio, 'Public Key Broadcast Encryption for Stateless Receivers', in
*ACM Workshop on Digital Rights Management*Springer, (2002), pp. 61-80.

16 Walid DOUCENE, 'Infrastructures À Clés Publiques Basées Sur La Technologie Blockchain', Mémoire MasterII, UNIVERSITE MOHAMED BOUDIAF-M'SILA-FACULTE MATHEMATIQUES ET DE L'INFORMATIQUE,(2019).

17 Explications Sur La Cryptographie', Tice-Education, (Publication : 16 mai 2020).

18 Mozilla included ca certificate list, https://wiki.mozilla.org/CA/Included_Certificates, consulté le : 20/02/2019.

19 Sécurité info,https://www.securiteinfo.com/cryptographie/pki.shtml, consulté le : 20/02/2019.
20 LeMaGiT, https://www.lemagit.fr/definition/Certificat-numerique, consulté le : 24/02/2019.
21 N.Chikouche, Sécurité informatique, polycopié de cours, 2018.