REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

N° d'ordre : .....

UNIVERSITE DE M'SILA

FACULTE DES MATHEMATIQUES ET DE L'INFORMATIQUE

Département d'Informatique

MEMOIRE de fin d'étude

Présenté pour l'obtention du diplôme de MASTER

Domaine : Mathématiques et Informatique
Filière : Informatique
Spécialité : Réseaux

Par: BRAHIMI AHLAM

SUJET

IMPLEMENTATION DES CRYPTOSYSTEMES BASES SUR LES CODES CORRECTEURS D'ERREURS

Soutenu publiquement le : / /2013 devant le jury composé de :

……	**Université de M'sila**	**Président**
Mr. CHIKOUCHE NOUREDDINE	**Université de M'sila**	**Rapporteur**
……	**Université de M'sila**	**Examinateur**
……	**Université de M'sila**	**Examinateur**

Promotion : 20 14

DEDICACE

Je dédie ce modeste travail à mes chers parents à qui je souhaite une longue vie.

A celle, dont le nom représente le premier mot que j'ai appris et qui reflète la plus grande sincérité de cette existence ; à celle qui a vécu pour être à mes côtés, partageant avec moi le meilleur et le pire, que DIEU le tout-puissant la garde :

A celui qui a été le soutien et la source de ma fierté et mon espoir mon cher père qu'ALLAH le garde.

A ceux avec qui j'ai partagé l'innocence de l'enfance, mes frères et sœurs :

Tarek – Khalil – Rania - Sohaib

A mes sœurs et les compagnes de mon parcours de vie :

Halima – Siham – Sabrina – Oumaima - Imen - Fatiha

A mes chères copines :

Nabila et mes collègues : Soumia –Hiba – hadjer –zahra – Amal-halima

A mes chères oncles A tous les cousins et cousines à ceux qui m'ont connue et aimée, que le destin a voulu que nos chemins se croisent, que le stylo n'a pu citer, mais qui sont dans mon cœur.

Au nom du DIEU le clément et le miséricordieux

Remerciements

Louange à DIEU qui m'a donné la foi, le courage et la patience de parvenir à finir ce modeste travail.

Je tiens à remercier :

- *Mon encadreur* ***Mr. CHIKOUCHE NOUREDDINE*** *qui m'a apporté une aide si précieuse. Je lui exprime ma gratitude pour sa grande disponibilité ainsi que pour sa compréhension et les encouragements qu'il m'a apportés.*
- ***Mr LAJLET LAHCEN*** *prof de math qui m'a guidé et conseillé.*
- *Mr BOU GHRARA SADEK .*
- *Tous les enseignants et professeurs, du primaire à l'université, qui sans eux, je n'aurais pu accéder à toute cette formation.*
- *A tous ceux qui ont contribué d'une manière ou d'une autre, de près ou de loin, à la réalisation de ce modeste travail.*

Introduction

Dans le domaine de la théorie de l'information, les codes sont utilisés pour détecter et corriger les erreurs de transmission. En effet, lorsqu'une source souhaite transmettre un message à un destinataire, elle envoie ce message au travers d'un canal dans lequel peuvent apparaître des erreurs dues à des bruits.

Ainsi le message que le destinataire reçoit peut avoir été modifié. Afin de protéger l'information de ce genre d'erreurs, on ajoute au message de la redondance avant de le transmettre dans le canal. A la réception, le décodeur tente de récupérer le message originel à partir du message reçu.

Pour corriger ces erreurs, on utilise le code correcteur des erreurs qui est basé sur des problèmes difficiles qui sont : Le problème du décodage (déterminer le mot du code le plus proche de m : message clair) et le problème de déterminer sa distance minimale.

Mceliece utilise la théorie des codes correcteurs d'erreurs à des fins Cryptographiques, et plus précisément pour un algorithme de chiffrement asymétrique.

Leur principe est : Alice envoyer un message contenant un grand nombre d'erreurs, erreurs que seul Bob sait détecter et corriger.

Niederreiter c'est une variante de McEliece reposant sur le problème de décodage par syndrome.

L'idée consiste à mettre l'information dans l'erreur et à envoyer un syndrome de cette erreur.

Niederreiter basé sur le Problème de la recherche des mots de poids inférieur à w(w :le poids de hamming) :

Trouver, s'il existe, un vecteur binaire e de longueur n et de poids inférieur ou égal à w tel que : eH = s.

Pour assurer l'authentification nous avons utilisé les schémas d'identification qui sont des systèmes d'identification à divulgation nulle de connaissance. Ce système est un Protocole cryptographique permettant à un fournisseur de preuve de convaincre un vérificateur de la validité d'un énoncé mais sans révéler aucune autre information que la véracité de cet énoncé.

Les schémas d'identification qui seront utilisés sont Stern, Veron et Cayrel-Veron-Al Yousfi .

Dans le premier chapitre de ce mémoire, nous présentons d'une façon générale des notions sur la cryptographie comme les mécanismes et les buts de la cryptographie, Ensuite nous passons aux cryptanalyse et les types de chiffrements. Pour le deuxième chapitre nous présentons des concepts mathématiques sur les codes correcteurs d'erreurs. Nous présentons les codes correcteurs d'erreurs, codes et distance de hamming, codes linéaires. Enfin nous présentons les codes BCH.

Dans le troisième chapitre nous exposons les Algorithmes de Chiffrement basés sur les codes de correcteurs d'erreurs McEliece, Niederreiter.

Le quatrième chapitre, comporte les trois schémas d'identification Stern, Veron, Cayrel-Veron-Al Yousfi. Dans le dernier chapitre, nous implémentons les systèmes de chiffrement à clé publique. Nous testons les performances selon le temps d'exécution et la capacité de correction de chaque algorithme, puis nous testons les performances selon le temps d'exécution pour chaque schéma d'identification. Enfin, nous comparons nos résultats.

Table des matières

Dédicace……... 4

Remerciement……. 5

1Introduction…….. 6

Table des matières…….. 7

1 Notions sur la cryptographie…….…… 10

1.1 Introduction…….…… 11

1.2 La cryptographie…….……. 11

1.2.1Mécanismes de la cryptographie…… 11

1.2.2 Les buts de la cryptographie…….. 11

1.3 La cryptanalyse…….. 12

1.4 Chiffrement symétrique…… 12

1.5 Chiffrement Asymétrique…….. 13

1.6 Chiffrement basé sur les codes de correcteurs d'erreur……. 14

1.7 Fonction de hachage…….. 15

1.8 Signature numérique…….. 15

1.9 Conclusion…… 16

2 Concepts mathématiques sur les codes de correcteurs d'erreur……... 17

2.1 Introduction……..... 18

2.1.1 Motivations……..... 18

2.1.2 Rappel……. 19

2.2 Les codes correcteurs……... 23

2.3 Codes et distance de Hamming……..... 24

2.4 Les codes linéaires…… 25

2.4.1 Définition…….. 25

2.4.2 Représentations matricielles……..... 25

2.4.2.1 Matrice génératrice……... 26

2.4.2.2 Matrice de contrôle……. 27

2.5 Décoder……...……. 28

2.5.1 Méthodes de décodage……..... 28
2.5.2 Capacité de correction…… 28
2.6 Codes BCH……..... 29
2.6.1 Définitions……..... 29
2.6.2 Equation clé…….. 30
2.6.3 Contexte de décodage……... 31
2.6.4 Equation clé……... 31
2.6.5 Les BCH binaires……...……... 32
2.7 Conclusion……...…….. 33
3 Algorithme de Chiffrement basé sur les codes correcteurs d'erreur……...……. 34
3.1Introduction……..….. 35
3.2 Cryptosystèmes basés sur des codes……..…….... 35
3.3 McEliece……. 35
3.3.1 Cryptosystème de McEliece……... 35
3.3.1.1 Cryptosystème de McEliece…….….. 36
3.3.1.2 Chiffrement……...…..... 37
3.3.1.3 Déchiffrement…….…….. 37
3.3.2 Paramètres du système…….. 38
3.3.3 Les avantages de McEliece……...….. 39
3.3.4 Les inconvénients de McEliece…….. 39
3.3.5Cryptanalyse du système de McEliece……..... 39
3.4 Niederreiter……..…… 40
3.4.1 Chiffrement……..……..… 41
3.4.2 Déchiffrement…….…… 41
3.4.3 Principe général du système de Niederreiter……...…..... 41
3.4.3.1 Problème de la recherche des mots de poids…….. 41
inférieur à *w* d'un coset
3.4.4 Paramètre……..…..... 41
3.5 Conclusion……..... 42
4 Schémas d'identification……...……. 43
4.1 Introduction…….……..…. 44
4.2 Preuve de connaissance à divulgation nulle……..…….. 44
4.2.1 Identification Zero-Knowledge…… 44

4.3 Schéma d'identification de stern…….. 45
4.3.1 Le schéma de base…….. 45
4.3.2 Le protocole de Stern généralisé…….... 46
4.3.3 Algorithme de schéma d'Identification Stern……... 47
4.4 Schéma d'identification de Véron…….. 48
4.4.1 Algorithme de Schéma d'Identification Véron…….. 49
4.5 Schéma d'identification de Cayrel-Veron-El Yousfi…….. 50
4.5.1 Le protocole de Cayrel-Véron-ElYousfi…….. 51
4.6 Autour de l'identification…….. 52
4.6.1 Matrice Random…….. 52
4.6.2 Matrice Quasi-cyclique……..... 52
4.6.3 Matrice Quasi-diadique……... 52
4.7 Conclusion……..... 52
5 Implémentation et test de performance……...… 53
5.1 Introduction……..... 54
5.2 Environnement de développement…….. 54
5.2.2 NetBeans…….. 54
5.2.3 JAVA……..... 54
5.3 Caractéristiques d'application…….. 55
5.3.1 Architecture de notre application……..... 55
5.3.2 Certain interface de notre application…….. 56
5.4 Implémentation et résultats expérimentaux…….. 58
5.4.1 Bibliothèques d'extensions relatives à la sécurité…….. 58
5.4.1.1 Flexi Provider…….. 59
5.4.1.2 Java Crypto API…….. 60
5.4.2 Calcul du temps d'exécution 60
5.4.3 Résultats expérimentaux…….. 61
5.4.3.1 Le temps d'exécution…….. 61
5.4.3.2 La capacité de correction…….. 63
5.6 Etude Comparative…….. 64
5.6.1 Comparaison les différents cryptosystemes…….. 64
5.6.2 Comparaison les différents schémas……... 65
5.7 Conclusion…….. 69

6 Conclusion générale……..... 70
Bibliographie……. 71

CHAPITRE 1

Notions sur la cryptographie

1 INTRODUCTION

Dans ce chapitre, nous proposons des notions sur la cryptographie : définition, mécanismes et buts de la cryptographie. Ensuite, il comportera d'autres définitions qui sont : la cryptanalyse et les trois types de chiffrements (symétriques, asymétriques et chiffrements basés sur les codes de correcteurs d'erreur).
Enfin, nous prendrons la fonction de hachage et la signature numérique.

2 La cryptographie

La *c*ryptographie est la science qui utilise les mathématiques pour le cryptage et le décryptage de données.
Elle vous permet ainsi de stocker des informations confidentielles ou de les transmettre sur des réseaux non sécurisés (tels que l'Internet), afin qu'aucune personne autre que le destinataire ne puisse les lire. Alors que la cryptographie consiste à sécuriser les données.

2.1 Mécanismes de la cryptographie

- Un algorithme de cryptographie ou un chiffrement est une fonction mathématique utilisée lors du processus de cryptage et de décryptage. Cet algorithme est associé à une clé (un mot, un nombre ou une phrase), afin de crypter le texte en clair.

  Avec des clés différentes, le résultat du cryptage variera également.

  La sécurité des données cryptées repose entièrement sur deux éléments : l'invulnérabilité de l'algorithme de cryptographie et la confidentialité de la clé.
- Un système de cryptographie est constitué d'un algorithme de cryptographie, ainsi que de toutes les clés et tous les protocoles nécessaires à son fonctionnement [10].

2.2 Les buts de la cryptographie

Confidentialité : mécanisme pour transmettre des données de telle sorte que seul le destinataire autorisé puisse les lire.
Intégrité : mécanisme pour s'assurer que les données reçues n'ont pas _été modifiées durant la transmission.

Authentification : mécanisme pour permettre d'identifier des personnes ou des entités et de certifier cette identité.

Non-répudiation : mécanisme pour enregistrer un acte ou un engagement d'une personne ou d'une entité de telle sorte que celle-ci ne puisse pas nier avoir accompli cet acte ou pris cet engagement [3].

3 La cryptanalyse

Est l'étude des informations cryptées, afin d'en découvrir le secret. La cryptanalyse Classique implique une combinaison intéressante de raisonnement analytique, d'application d'outils mathématiques, de recherche de modèle, de patience, de détermination et de chance. Ces cryptanalyses sont également appelés des pirates.

4 Chiffrement symétrique

Cryptage de clé secrète ou de clé symétrique, une seule clé suffit pour le cryptage et le décryptage.

La norme de cryptage de données (DES) est un exemple de système de cryptographie conventionnelle largement utilisé par le gouvernement fédéral des Etats-Unis.

Le cryptage conventionnel (de clé symétrique) comporte des avantages. Il est très rapide. Mais, il s'avère particulièrement utile pour les données véhiculées par des moyens de transmission sécurisés. Toutefois, il peut entraîner des coûts importants en raison de la difficulté à garantir la confidentialité d'une clé de cryptage lors de la distribution.

Un expéditeur et un destinataire souhaitant communiquer de manière sécurisée à l'aide du cryptage conventionnel doivent convenir d'une clé et ne pas la divulguer. S'ils se trouvent à des emplacements géographiques différents, ils doivent faire confiance à un coursier, au téléphone de Batman ou à tout autre moyen de communication sécurisé pour éviter la divulgation de la clé secrète lors de la transmission. Toute personne interceptant la clé lors d'un transfert peut ensuite lire, modifier et falsifier toutes les informations cryptées ou authentifiées avec cette clé. De la norme de cryptage de données DES au code secret de Jules César, la distribution des clés reste le problème majeur du cryptage conventionnel. Autrement

dit, comment faire parvenir la clé à son destinataire sans qu'aucune personne ne l'intercepte ?

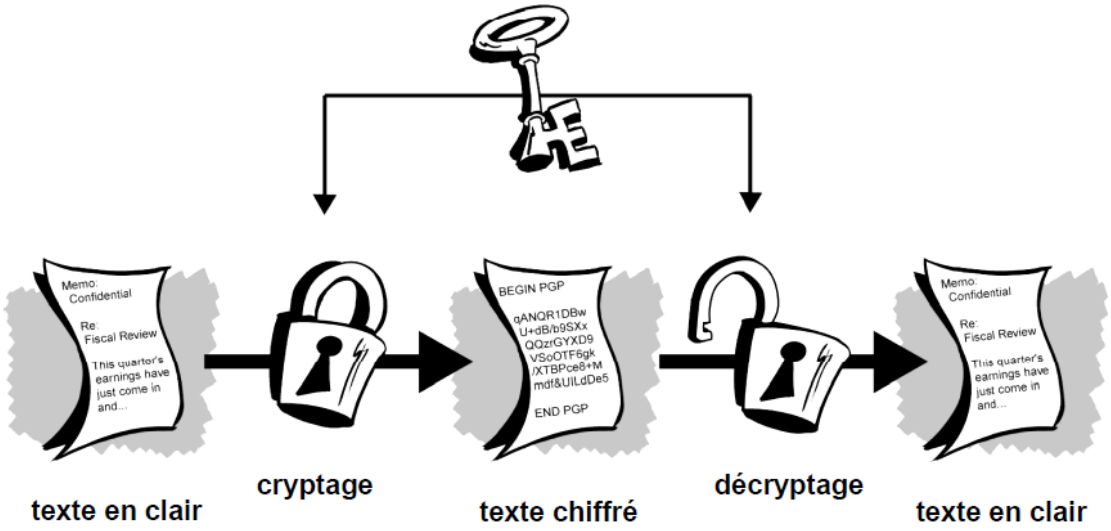


Figure 1.1 Cryptage de clé secrète

5 Chiffrement Asymétrique

La cryptographie de clé publique est un procédé asymétrique utilisant une paire de clés pour le cryptage : une clé publique qui crypte des données et une clé privée ou secrète correspondante pour le décryptage. Vous pouvez ainsi publier votre clé publique tout en conservant votre clé privée secrète. Tout utilisateur possédant une copie de votre clé publique peut ensuite crypter des informations que vous êtes le seul à pourvoir lire. Même les personnes que vous ne connaissez pas personnellement peuvent utiliser votre clé publique.

Il est impossible de deviner la clé privée à partir de la clé publique. Tout utilisateur possédant une clé publique peut crypter des informations, mais est dans l'impossibilité de les décrypter. Seule la personne disposant de la clé privée correspondante peut les décrypter.

La cryptographie de clé publique présente un avantage majeur : en effet, elle permet d'échanger des messages de manière sécurisée sans aucun dispositif de sécurité. L'expéditeur et le destinataire n'ont plus besoin de partager des clés secrètes via une voie de transmission sécurisée. Les communications impliquent uniquement l'utilisation de clés publiques et plus aucune clé privée n'est transmise ou partagée. Elgamal (d'après le nom de son inventeur, Taher Elgamal), RSA (d'après le nom de

ses inventeurs, Ron Rivest, Adi Shamir et Leonard Adleman), Diffie-Hellman (également d'après le nom de ses inventeurs) et DSA, l'algorithme de signature numérique (élaboré par David Kravitz), sont des exemples de systèmes de cryptographie de clé publique [10].

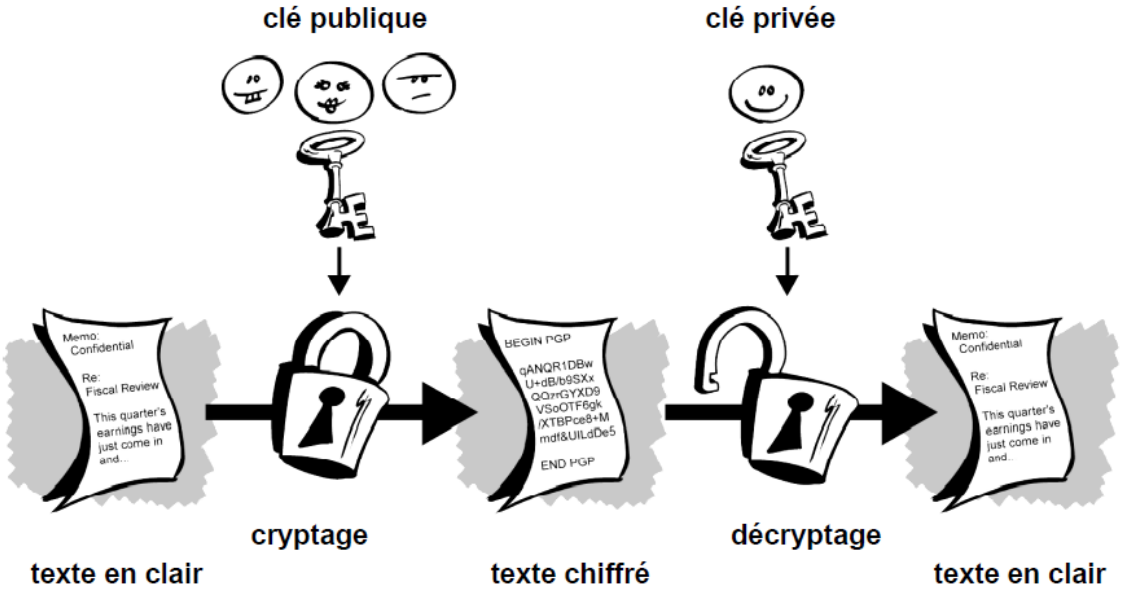


Figure 1.2 Cryptage de clé publique

6 Chiffrement basé sur les codes de correcteurs d'erreur

Le principe est simple, il s'agit d'ajouter de la redondance dans le message transmis. On obtient ainsi un message plus long, mais dont l'information excédentaire peut être exploitée pour détecter voire corriger des erreurs de transmission.

Les codes correcteurs sont utilisés ainsi dans toutes les télécommunications, mais aussi dans le stockage de données (gravure de Cd-Rom, par exemple) ou encore dans les identifiants personnels (numéro de sécurité social, de comptes bancaires, etc.).

Le domaine de la théorie des codes a donc connu un très fort développement ces soixante dernières années. Ce développement a donné naissance à des analyses mathématiques et algorithmiques poussées des opérations possibles sur les codes correcteurs. Notamment, il semble clair que le décodage d'un code aléatoire est un problème algorithmique difficile.

Pour mettre au point un système cryptographique, il faut un problème algorithmique difficile. McEliece à présente dans l'idée d'utiliser le décodage comme problème difficile sous-jacent à un système cryptographique.

Ainsi, les codes correcteurs d'erreurs, destinés à l' origine à clarifier des messages Transmis, peuvent être utilisés au contraire pour les chiffrer [4].

7 Fonction de hachage

Le hachage d'un message consiste en l'application d'une fonction mathématique qui permet d'en créer un condensé. Ce dernier est évidemment de taille beaucoup plus petite que le message lui-même si bien que la fonction de hachage n'est pas biunivoque. Il n'est pas non plus possible de reconstituer le message à partir du condensé.

La technique de hachage est utilisée pour préserver l'intégrité d'une information en envoyant cette information accompagnée de son condensé, ce dernier étant éventuellement chiffré.

La fonction de hachage est telle qu'il est pratiquement impossible statistiquement de produire deux messages dont les condensés seraient identiques.

Le message et son condensé sont donc liés : on ne peut pas modifier l'un sans devoir modifier l'autre.

L'association du hachage et du chiffrement permet de réaliser la fonction de signature décrite ci-après. C'est la technique essentielle utilisée par l'infrastructure à gestion de clés.

8 Signature numérique

La notion de signature numérique d'un message est fondamentale dans l'authentification de l'origine d'une transaction ainsi que dans sa garantie de non-répudiation. Elle s'obtient en associant une opération de hachage et une opération de chiffrement asymétrique, comme indiqué sur la figure 3 et le texte ci-après.

1. Création par l'émetteur d'un condensé du message par une opération de hachage

2. Chiffrement asymétrique du condensé par la clé privée de l'émetteur. Ceci constitue la signature.

3. Envoi des deux informations (message et signature) au destinataire.

Le destinataire qui reçoit ces deux informations doit, pour vérifier la signature, procéder ainsi.

4. Calcul à nouveau du condensé du message par le même algorithme que celui utilisé par l'émetteur.

5. Déchiffrement de la signature en utilisant la clé publique de l'émetteur.

Cela permet de reconstituer le condensé créé par l'émetteur.

6. Comparaison des deux informations condensées ainsi obtenues.

Si elles sont identiques, seul l'émetteur (le possesseur de la clé privée) a pu envoyer ce message.

L'opération de vérification de la signature suppose que le destinataire possède la clé publique de l'émetteur. La plupart du temps, ce dernier peut l'envoyer, incluse dans un certificat, en même temps que le message.

Nous voyons donc ici l'importance d'une autorité externe fiable permettant de certifier la clé publique fournie par l'émetteur d'un message. C'est précisément le rôle de la fonction de certification que de fournir la preuve que la clé publique est bien celle appartenant à l'émetteur du message. Cette preuve est le certificat signé par cette autorité externe [9].

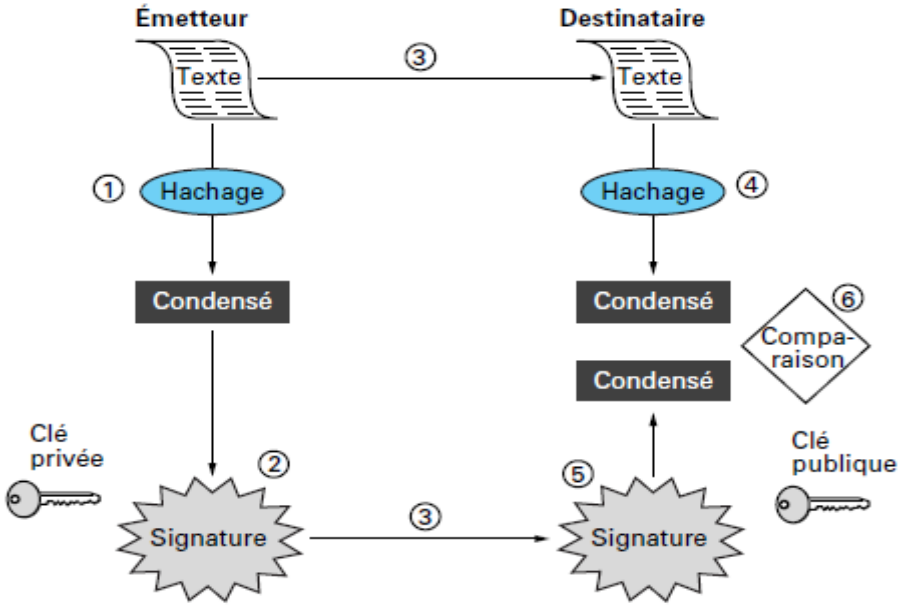


Figure 1.3 Principe du codage de la signature : association
Du hachage et du chiffrement asymétrique

9 Conclusion

Dans ce chapitre, nous avons proposé des concepts sur la cryptographie. Ensuite nous avons passé aux types de chiffrements.

Enfin, nous prendrons la fonction de hachage et la signature numérique.

CHAPITRE 2

Concepts mathématiques sur les codes correcteurs d'erreurs

1 INTRODUCTION

Dans ce Chapitre nous proposons des concepts mathématiques sur les codes correcteurs des erreurs**.** Nous commençons par des motivations et un rappel. Ensuite nous passons aux codes correcteurs des erreurs, codes et distance de hamming, codes linéaires. Enfin nous présentons les codes BCH.

1.1 Motivations

La plupart des cryptosystèmes à clé publique utilisés sont basés sur des problèmes de théorie des nombres. Il est important de connaître des systèmes alternatifs efficaces en pratique (réseaux, systèmes multivariés, codes correcteurs d'erreurs, fonctions de hachage. .*)*.

Les avantages de la cryptographie basés sur les codes correcteurs d'erreurs: *a priori* sûre face à l'ordinateur quantique, des problèmes NP-complets bien connus, rapides et faciles à implanter.

Les désavantages de la cryptographie basés sur les codes correcteurs d'erreurs : grande taille de clé publique (des centaines de milliers de bits . . .*)*.

Récemment, les systèmes basés sur les codes correcteurs d'erreurs ont été présentés avec de petites tailles de clés (aux alentours de 30 000 bits pour le chiffrement) [17].

La figure suivante présente un schéma général de communication :

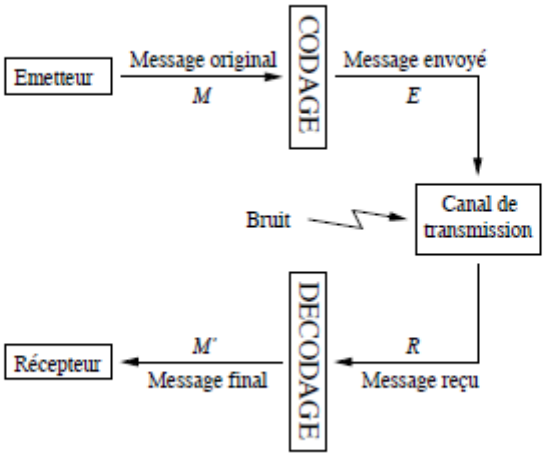


Figure2.1 Schéma général de communication

1.2 Rappel

- K corps fini= k corps commutatif + card k fini

(k , +, .) Corps commutatif :

1) (k, +) groupe commutatif

   + associative (x + y) + z = x + (y + z).

   + commutative x + y = y + x.

   + admet un élément neutre '' 0'' : x + 0 =0 + x =x.

   Chaque élément x admet une symétrie (-x) : x + (-x) = (-x) + x = 0.

2) . associative x .(y . z) =(x . y) . z.

   . commutative x . y = y . x.

   . admet un élément neutre x . 1 = 1 . x = x.

3) . distributive par rapport à + : x . (y + z) = xy + xz.

   (x + y) . z = xz + yz.

4) ∀ x ∈ k, x ≠ 0, x inversible : ∃ y ∈ k : xy = yx = 1.

- P premier $\mathbb{Z}/_{p\mathbb{Z}}$ = { $\bar{0}$, $\bar{1}$,……, $\overline{p-1}$}

Exemple 1. $\mathbb{Z}/_{5\mathbb{Z}}$ = { $\bar{0}$, $\bar{1}$, $\bar{2}$, $\bar{3}$, $\bar{4}$}

$\bar{3} + \bar{4} = \overline{3+4} = \bar{7} = \bar{2}$

$\bar{3} \,.\, \bar{4} = \overline{3.4} = \overline{12} = \bar{2}$

$\overline{-4} = \bar{1}$

$\overline{-8} = \overline{-3} = \bar{2}$

- $\mathbb{F}_p$ = ($\mathbb{Z}/_{p\mathbb{Z}}$,+, .) corps commutatif

  ⇕

  P premier corps fini (p) élément

  k corps fini ,card k = q

  k= $\mathbb{F}_p$ = field of q elements

  Suite (Exmple1):

  P = 5 $\bar{1} \,.\, \bar{1} = \bar{1}$ $\bar{1}^{-1} = \bar{1}$

  $\bar{2} \,.\, \bar{3} = \bar{1}$ $\bar{2}^{-1} = \bar{3}$

  $\bar{3} \,.\, \bar{2} = \bar{1}$ $\bar{3}^{-1} = \bar{2}$

  $\bar{4} \,.\, \bar{2} = \bar{1}$ $\bar{4}^{-1} = \bar{4}$

  $\mathbb{Z}/_{4\mathbb{Z}}$ = { $\bar{0}$, $\bar{1}$, $\bar{2}$, $\bar{3}$ } n'est pas un corps

  $\bar{2} \,.\, \bar{0} = \bar{0}$

  $\bar{2} \,.\, \bar{1} = \bar{2}$

$\bar{2} \cdot \bar{2} = \bar{4} = \bar{0}$

$\bar{2} \cdot \bar{3} = \bar{6} = \bar{2} \longrightarrow \bar{2}$ non-inversible

Exemple 2. $\{0,1\} \longrightarrow \{000,111,101,011,\ldots\}$

$\mathbb{F}_2^1 \longrightarrow \mathbb{F}_2^3$

Rq : $\mathbb{Z}/_{2\mathbb{Z}} = \{\bar{0}, \bar{1}\} = \{0,1\}$

$\bar{1}+\bar{1}= \bar{0}$ $\quad 1 + 1 = 0$

$1 + 0 = 1$

$1 \cdot 1 = 1$

$-1 = 1$

$\mathbb{F}_2^k \xrightarrow{\varphi} \mathbb{F}_2^n$

mot $a_1,a_2,\ldots\ldots,a_k \longrightarrow x_1,x_2,\ldots\ldots,x_n$

vecteur $(a_1,a_2,\ldots\ldots,a_k) \longrightarrow (x_1,x_2,\ldots\ldots,x_n)$

φ : Le codage (φ injective)

Exemple 3. $\mathbb{F}_2^2 \xrightarrow{\varphi} \mathbb{F}_2^4$

$a_1,a_2 \longrightarrow x_1,x_2,x_3,x_4$

$$\begin{cases} x_1 = a_1 \\ x_2 = a_2 \\ x_3 = a_1 + a_2 \\ x_4 = a_2 \end{cases}$$

$01 \longrightarrow 0111$

$00 \longrightarrow 0000$

$10 \longrightarrow 1010$

$11 \longrightarrow 1101$

$|\mathbb{F}_2^n| = |2^n|$

$C= \{0111,0000,1010,1101\} \subseteq \mathbb{F}_2^4$ le code binaire de longueur 4.

Définition : 1- un code binaire C de longueur n est une partie non vide de $\mathbb{F}_2^n$

2- un code linéaire de longueur n est un sous espace vectoriel de $\mathbb{F}_2^n$

C est un code linéaire de longueur n(binaire sur $\mathbb{F}_2^n$)

$\Updownarrow$

$\forall\ x = x_1\ x_2\ \ldots\ldots\ldots x_n$ et $y = y_1\ y_2\ldots\ldots\ldots y_n \in C$ (sous espace vectoriel)

$x+y \in C$

Cas général (binaire ou non binaire)

C est un code linéaire de longueur n

1- $\forall x,y \in C \Rightarrow x+y \in C$

2- $\forall \lambda \in \mathbb{F}_q, \forall x \in C, \lambda\ x \in C$ } sous espace vectoriel

$C = \{0111,0000,1010,1101\} \subseteq \mathbb{F}_2^4$

C : code binaire de longueur 4.

$$\begin{array}{cccc} 0111 & 0111 & 0111 & 1010 \\ +\ 0000 & +\ 1010 & +\ 1101 & +\ 1101 \\ \hline 0111 & 1101 & 1010 & 0111 \end{array}$$

C code binaire

Exemple 4. $D = \{010,110,000\} \subseteq c$

$$\begin{array}{r} 010 \\ +\ 110 \\ \hline 100 \end{array} \quad \notin D$$

D code non linéaire

- C code linéaire $\Rightarrow$ C sous espace vectoriel de $\mathbb{F}_2^n$
- ß = { $v_1,v_2,\ldots,v_k$} base de l'espace vectoriel V, $B \subseteq V$

Si

1) ß linéairement indépendant (libre)

$\alpha_1v_1+ \alpha_2v_2+\ldots+ \alpha_nv_n = \vec{0} \Rightarrow \alpha_1= \alpha_2 =\ldots\ldots\ldots= \alpha_n = 0$

2) ß engendre V

$\forall v \in V : v = \lambda_1v_1+ \lambda_2v_2 + \ldots..+ \lambda_nv_n \qquad \lambda_i \in \mathbb{F}_q$

(v = combinaison linéaire de B)

$\mathbb{R}^2$ {(1,0),(0,1)} base de $\mathbb{R}^2$

- $\alpha_1(1,0) + \alpha_2(0,1) = (0,0) \leftrightarrow (\alpha_1, \alpha_2) = (0,0) \leftrightarrow \alpha_1= \alpha_2 = 0$
- $v = (x,y) \in \mathbb{R}^2$

(x,y) = (x,0) + (0,y)

v = x(1,0) + y(0,1)

$v = x\ \vec{e}_1 + y\ \vec{e}_2$

Le nombre des vecteurs formant une base est appelé la dimension de l'espace vectoriel V

- C code linéaire :

$\Rightarrow$ soit $\{g_1,g_2,\ldots..,g_k\}$ une base de C

$$\left.\begin{array}{l} g_1 = (g_{11}, g_{12}, \dots, g_{1n}) \\ g_2 = (g_{21}, g_{22}, \dots, g_{2n}) \\ \vdots \quad \vdots \quad \vdots \quad \ddots \quad \vdots \\ g_k = (g_{k1}, g_{k2}, \dots, g_{kn}) \end{array}\right\} \in C$$

$$G = \begin{bmatrix} g_{11} & g_{12} & \dots & g_{1n} \\ g_{21} & g_{22} & & g_{2n} \\ \vdots & & \ddots & \vdots \\ g_{k1} & g_{k2} & \dots & g_{kn} \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_k \end{bmatrix}$$

1) G de type k×n

   K= dim C , n = la longueur

2) Rang G = k

3) $C = \{ mG : m =(a_1,a_2,\dots\dots,a_k) \in \mathbb{F}_2^k \}$

   $x \in C \leftrightarrow x=mG, m \in \mathbb{F}_2^k$

Exemple 5.

$C= \{0111,0000,1010,1101\} \subseteq \mathbb{F}_2^4$ le code binaire de longueur n=4.

Une base ?

G =?

$0111 = 0. g_1 + 1. g_2$

$0000 = 0. g_1 + 0. g_2$

$1010 = 1. g_1 + 0. g_2$

$1101 = 1. g_1 + 1. g_2$

$g_1 = 1010 = (1,0,1,0)$

$g_2 = 0111 = (0,1,1,1)$

$\alpha_1 g_1 + \alpha_2 g_2 = (0,0,0,0)$

$(\alpha_1,0, \alpha_1,0) + (0, \alpha_2, \alpha_2, \alpha_2) = (0,0,0,0)$

$(\alpha_1, \alpha_2, \alpha_1+ \alpha_2, \alpha_2) = (0,0,0,0) \Rightarrow \alpha_1 = \alpha_2 = 0$

{ g_1, g_2 }base de C → $G = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$ Matrice génératrice de C

Autre façon :

$$G = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$ Matrice génératrice

C code linéaire binaire : dim C = 2 , n = la longueur = 4 , $C \subseteq \mathbb{F}_2^4$.

C = {mG : m Є $\mathbb{F}_2^2$ }= { a_1 a_2 G : a_1,a_2 = 0,1} = le code

$\mathbb{F}_2^2$ = {00,01,10,11}= l'ensemble des messages

C = {00G,01G,10G,11G}

00G = (0,0) $\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$ = (0,0,0,0) = 0 0 0 0

01G = (0,1) $\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$ = (0,1,1,1) = 0 1 1 1

10G = (1,0) $\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$ = (1,0,1,0) = 1 0 1 0

11G = (1,1) $\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$ = (1,1,0,1) = 1 1 0 1

C = {0000,0111,1010,1101}

C = { $\alpha_1 g_1 + \alpha_2 g_2 + \ldots\ldots + \alpha_k g_k : \alpha_1, \alpha_2, \ldots\ldots, \alpha_k = 0,1$}

2 Les codes correcteurs

Les codes correcteurs ont été introduits pour corriger les erreurs de transmission ou de lecture de données numériques, ou les erreurs survenant au cours de leur inscription sur un Support physique (bande, CD) ou encore lorsque les données subissent une altération sur le Support de stockage. Voici quelques domaines où ils sont appliques :

- Transmissions spatiales
- Minitel
- Codes-barres
- Disque compact et DVD
- Communications par internet.

Le schéma suivant illustre la chaîne d'actions qui permet à une source de transmettre de l'information à un destinataire [17]

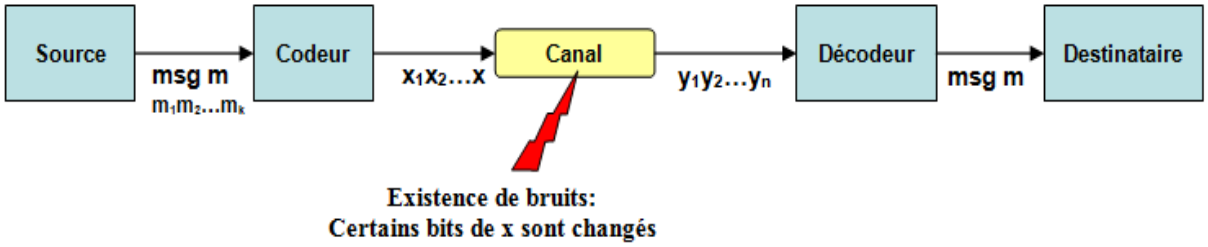


Figure2.2 la chaîne d'actions qui permet à une source de transmettre de l'information à un destinataire

3 Codes et distance de Hamming

Les messages transmis sont supposés découpes en blocs (ou mots) de longueur n écrits avec l'alphabet {0,1}. Un code (binaire) est un sous-ensemble C de l'ensemble $\{0,1\}^n$ de tous les mots possibles. On dit que n est la longueur de C.

La distance de Hamming entre deux mots $x = (x_1,\ldots., x_n)$ et $y = (y_1,\ldots., y_n)$, que l'on notera d(x,y), est le nombre d'indices i tels que xi 6= yi. C'est bien une distance sur $\{0,1\}^n$. La distance minimum du code C'est le minimum des d(x,y) pour x et y des mots différents de C (on suppose que C a au moins 2 mots !).

On la notera toujours d.

Exemple 6. Considère $C = \{c_0, c_1, c_2, c_3\}$ avecc0 = (00000); c1 = (10110);

c2 = (01011); c3 = (11101): C'est un code de longueur 5 et de distance d = 3.

Le mot c ϵ C est émis et, après d'éventuelles erreurs de transmission, le mot r ϵ $\{0,1\}^n$ Est reçu. On décode le mot r selon le principe du maximum de vraisemblance, c.-à-d. qu'on le décode comme un mot de C à distance minimum de r. On dit que C'est t-correcteur (ou corrige t erreurs) quand toute erreur portant sur au plus t bits est corrigée correctement. On voit donc que le code C est t-correcteur si et seulement si les boules fermées (dans $\{0,1\}^n$ muni de la distance de Hamming) de centres les éléments de C et de rayon t sont disjointes, ou encore si et seulement si la distance minimum d de C vérifie $d \geq 2t + 1$.

Il est souvent difficile de calculer la distance minimale et plus encore de décoder un mot sans structure additionnel c'est pourquoi on préfère travailler avec des codes linéaires [6].

4 Les codes linéaires

Codes correcteurs pour lesquels la redondance dépend linéairement de l'information ;

Peuvent être définis par une matrice génératrice G de taille $k \times n$ [17] :

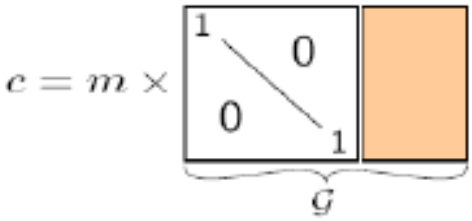


Figure2.3 La définition de code correcteur par une matrice génératrice

4.1 Définition

Un code binaire C est linéaire si la somme de deux mots quelconques du code est encore un mot du code :

$w1, w2 # C, w1 + w2 # C

Un code linéaire est donc un sous-espace vectoriel de $(\mathbb{Z}/2\mathbb{Z})$ *n*. En particulier le nombre de mots du code est 2p, ou p est la dimension de C.

Remarque : dans un code linéaire, le mot 0n est toujours un mot du code :

En effet si x est un mot du code quelconque, alors x + x = 0n est aussi un mot du code [8].

4.2 Représentations matricielles

La détermination du mot de code associé à un bloc, telle que nous venons de la décrire, se ramène à un produit de matrices ; voyons comment.

Un mot binaire peut être représenté par une matrice ligne dont les coefficients sont les bits de ce mot. Pour économiser les notations, nous donnerons le même nom à la matrice ligne et au mot binaire qu'elle représente.

Exemple 7. Le mot binaire b=01001 est représenté par matrice ligne : b= (0 1 0 0 1)

On appelle **matrice génératrice** du codage et on note G, la matrice à k lignes et n colonnes obtenue en écrivant l'un au-dessous de l'autre et dans l'ordre, les mots de code des blocs b_i de la base canonique.

Parce que le codage est systématique, les k bits d'information sont écrits en premier, sans en changer l'ordre, et les bits de contrôle viennent ensuite. Cette particularité fait que G apparait comme la juxtaposition de deux matrices (figure2.4) :

à gauche I_k, la matrice identité d'ordre k, et à droite une matrice à k lignes et r colonnes, notée P, qu'on appelle la **matrice de parité** et qui est constituée des bits de contrôles des mots de code $\varphi\ (b_i)$ [13]**.**

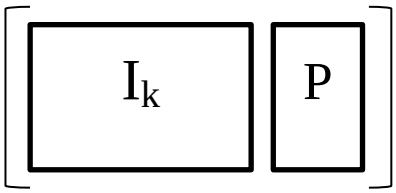
G= 

Figure2.4 La matrice génératrice sous forme systématique

4.2.1 Matrice génératrice

On peut se donner un sous-espace vectoriel (et donc un code) par une base. Soit C un code linéaire. Une matrice génératrice de C est une matrice dont les lignes forment une base de C. Une matrice génératrice G est donc de taille k × n et de rang k. Si m est un vecteur ligne de $|F_2^k$, le produit mG est un mot du code C et l'application $m \rightarrow mG$ est un isomorphisme de $|F_2^k$ sur C (que l'on peut voir comme une opération de codage). Si la matrice G est de la forme (I, P), on dit que le codage est systématique. Les k premiers bits d'un mot de code portent l'information (on y recopie le vecteur de$|F_2^k$), les n-k suivants sont de la redondance.

Exemple 8. La matrice

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

Est une matrice génératrice pour le code (00000), (10000), (11010), (11101), (01010), (01101), (00111), (10111).

On dit que deux codes linéaires de même longueur sont équivalents si l'un s'obtient à Partir de l'autre par une permutation des coordonnées. On peut vérifier que deux codes Équivalents ont même type. De plus tout code est équivalent à un code donné par un codage Systématique.

4.2.2 Matrice de contrôle

On peut aussi se donner un sous-espace vectoriel par un système d'équations indépendantes. Soit C un code linéaire. Une matrice de contrôle de C est la matrice d'un système d'équations linéaires homogènes indépendantes dont l'espace des solutions est C. Autrement dit, une matrice de contrôle H est de taille (n-k) ×n et de rang n-k, et C = $\{x \in \mathbb{F}_2^k, H^t x = 0\}$. Si C est donné sous forme systématique par la matrice génératrice G = (I_k , P), alors on peut prendre comme matrice de contrôle H = $(-{}^t P; I_{n-k})$ (le signe - est superflu en caractéristique 2).

Supposons que c ϵ C est le mot du code envoyé et r ϵ $\mathbb{F}_2^k$ le mot reçu.

La différence e = r-c est le vecteur d'erreur. Son poids w(e) est le nombre de bits erronés dans le mot reçu. Soit H une matrice de contrôle de C. Le syndrome du mot reçu r est le vecteur s ϵ $\mathbb{F}_2^k$défini par ${}^t s = H_r^t = H_e^t$.

Le syndrome est nul si et seulement si r ϵ C. Le syndrome définit un isomorphisme du quotient $\mathbb{F}_2^k$/C sur$\mathbb{F}_2^{n-k}$. Si le syndrome est non nul, on corrige le mot reçu r en appliquant le principe du maximum de vraisemblance : on soustrait à r un mot de poids minimum dans sa classe modulo C, c.-à-d. un mot de poids minimum parmi ceux ayant même syndrome que r. Dans le cas ou w(e) est strictement inferieur a d=2, alors e est l'unique mot de poids minimum dans la classe de r modulo C et on récupère bien le mot de code émis.

Remarque : La matrice de contrôle peut être vue comme la matrice génératrice du code dual $C^{\perp} = \{y \in |F_2^k, \forall\ c \in C, y.c = 0\}$ ou. Est le produit scalaire usuel.

Proposition : Soit H une matrice de contrôle du code C. La distance minimum d de C est caractérisée par les propriétés suivantes :

- ✓ d - 1 colonnes de H sont toujours linéairement indépendantes.
- ✓ Il y a un système de d colonnes de H qui est lié.

Exemple 9. Donnons un exemple de décodage par syndrome. Soit C le code donné par la matrice de contrôle

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

On calcule tout d'abord des représentants de poids minimal (appelé leader) pour chacune des classes $\mathbb{F}_2^6$ =C ainsi que le syndrome associé

Leader	Syndrome
000000	000
100000	110
010000	101
001000	011
000100	100
000010	010
000001	001
100001	111

Table 2.1 Leader et le syndrome

Supposons que u = 100011 est reçu. Son syndrome est $H^t u$ = 101. Pour décoder u il faut donc lui soustraire 010000 [6].

5 Décoder

L'émetteur envoie $c = mG$, mais le receveur reçoit $c' = c + e$.

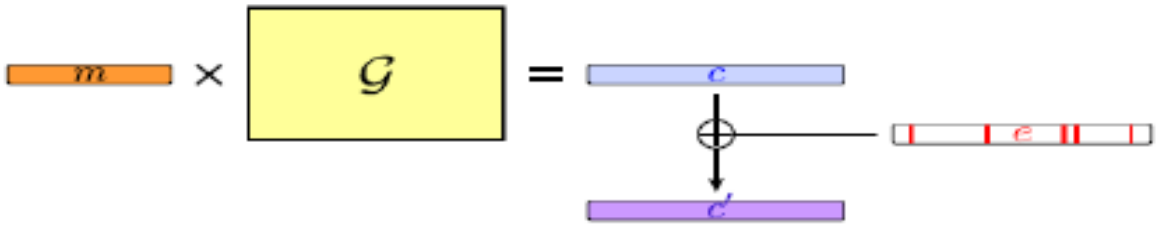


Figure 2.5 La communication entre l'émetteur et le récepteur

Décoder consiste à retrouver c à partir de c'.

Généralement, nous souhaitons un décodage à maximum de vraisemblance trouver le mot de code c le plus proche du mot reçu c'.

5.1 Méthodes de décodage

• Recherche exhaustive du vecteur d'erreur : choisir e de petit poids, calculer $c' - e$ et vérifier qu'il est dans le code.

• Recherche exhaustive de mots de code : calculer $c' - mG$ pour tous les m possibles et vérifier son poids.

→ Ces deux méthodes sont exponentielles.

5.2 Capacité de correction

Du code C de paramètres $[n, k, d]_q$ est l'entier t tel que $t = [\frac{d-1}{2}]$.

Pourquoi [17]?

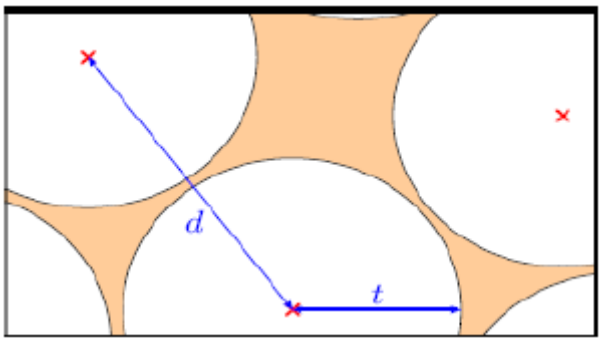


Figure 2.6 Distance minimale d et capacité de correction t

6 Codes BCH

Indépendamment, le mathématicien français Alexis Hocquenghem dans et les mathématiciens indiens Raj Chandra Bose et Dwijendra Chaudhuri ont introduit les codes appelés aujourd'hui BCH.

Les auteurs sont arrivés à la définition des codes BCH, en s'intéressant à des sous-codes des codes de Hamming.

6.1 Définitions

- Définition1

  (Code cyclique). Soit C un code de longueur n. Le code C est un code cyclique si et seulement si

  $c \triangleq (c_1, \ldots, c_n) \in C \Rightarrow (c_n, c1, \ldots, c_{n-1}) \in C$:

- Notation
- Comme $\mathbb{F}_{q^m}^n$ est isomorphe en tant qu'espace vectoriel à l'espace vectoriel des polynômes univariés de $\mathbb{F}_{q^m}$ de degré strictement inférieur à n, on a pour tout $\alpha \in \mathbb{F}_{q^m}$

  $e\vartheta_\alpha : \mathbb{F}_{q^m}^n \rightarrow \mathbb{F}_{q^m}[X] \rightarrow \mathbb{F}_{q^m}$

  $\vartheta = (\vartheta_1, \vartheta_2, \ldots, \vartheta_n) \rightarrow \sum_{i=1}^{n} \vartheta_i X^{i-1} \rightarrow \sum_{i=1}^{n} \vartheta_i \alpha^{i-1}$

- Définition2

  (Code BCH - Code d'annulation). Soient q le cardinal du corps, N, m,δ,b quatre entiers tels que pgcd (q, n) = 1, m le plus petit entier tel que $q^m \equiv 1$

mod n et α une racine primitive ne de l'unite sur$\mathbb{F}_{q^m}$. Alors le code BCH est défini par $C_{BCH} = \{\vartheta \in \mathbb{F}_q^n : \forall_i \in \{0,\ldots., \delta - 2\}, e\vartheta_{\alpha^{b+i}}(\vartheta) = 0\}$.

- Lemme
  Soient C un code BCH et H la matrice définie par :

$$H=\begin{pmatrix} 1 & \alpha^b & (\alpha^b)^2 & \cdots & (\alpha^b)^{n-1} \\ 1 & \alpha^{b+1} & (\alpha^{b+1})^2 & \cdots & (\alpha^{b+1})^{n-1} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \alpha^{b+\delta-2} & (\alpha^{b+\delta-2})^2 & \cdots & (\alpha^{b+\delta-2})^{n-1} \end{pmatrix}$$

Alors la matrice constituée des éléments de H développés en colonne sur $\mathbb{F}_q$ est une matrice de parité de C.

- Définition3 (Code BCH primitif au sens strict).
  Avec les notations précédentes, le code BCH défini est appelé un code BCH au sens strict si b = 1 ; il est dit primitif si $n = q^m - 1$.
- Définition4
  (Polynôme générateur). Avec les notations précédentes, on appelle polynôme générateur d'un code BCH le polynôme $g(X) \in \mathbb{F}_q[X]$ défini par
  $g(X) \triangleq \text{ppcm}\ (m_b(X),\ldots.., m_{b+\delta-2}(X))$, où $m_i(X)$ est le polynôme minimal deα^i.
- Proposition1
  Soit un code BCH de polynôme générateur g(X). Alors une matrice génératrice du code est

$$G \triangleq \begin{pmatrix} g_0 & g_1 & \cdots & g_{\deg(g)} & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & \cdots & g_{\deg(g)} & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \cdots & \ddots & 0 \\ 0 & \cdots & 0 & g_0 & g_1 & \cdots & g_{\deg(g)} \end{pmatrix}$$

Où g_i est le coefficient monomial de degré i de g(X).

- Proposition2

Soit C un code BCH de polynôme générateur g(X) et de distance construiteδ, alors dim(C) = n-deg(g) et la distance minimale a pour minoration $d_C \geq \delta$.

6.2 Equation clé

L'équation qu'on présente ici, appelée équation clé, permet de mettre en relation des polynômes qui caractérisent l'erreur et le syndrome. En résolvant cette équation, on est alors en mesure de déterminer l'erreur et ainsi le mot de code original.

Quelques algorithmes de décodage résolvent cette équation pour retrouver le mot de code transmis par exemple la méthode d'Euclide et celle de Berlekamp-Massey.

6.3 Contexte de décodage

Soient C un $[n, k]_q$-code BCH au sens strict et H une matrice de parité de C. Supposons qu'on souhaite transmettre le mot de code c Є C et que, durant la transmission, une erreur provoquée par le canal se produit.

Le récepteur est alors en possession du mot reçu $y \triangleq c + e$.

Avec y, e Є $\mathbb{F}_q^n$ et c 2 C. Le problème est de retrouver c à partir de la connaissance du code C et du mot reçu y. Si on applique la fonction syndrome associée à H sur y, on obtient :

$$S_H(y) = yH^t.$$
$$= (c + e)H^t.$$
$$= eH^t.$$

D'après la définition de la matrice de parité, H est une représentation matricielle d'une application linéaire qui a pour noyau le code, en appliquant la fonction syndrome à un mot reçu y, on obtient alors de l'information uniquement sur l'erreur.

De plus, en supposant que le poids de e soit inférieur à t la fonction syndrome est injective, On est donc en mesure de déterminer l'unique mot de code c à distance t de y.

6.4 Equation clé

L'erreur e est un élément de $\mathbb{F}_q^n$, donc on peut l'écrire sous la forme vectorielle On note w le poids de l'erreur e, c'est-à-dire $Supp(e) = \{i_1, \ldots, i_w\}$.

Pour alléger les notations et pour également respecter la présentation standard de l'équation de clé, on note $\forall_j$Є $\{1, \ldots\ldots, w\}$, $x_j \triangleq \alpha^{i_{j-1}}$ et $z_j \triangleq e_{i_j}$.

On définit ensuite deux polynômes importants : le polynôme localisateur Λ et le polynôme évaluateur L de la manière suivante :

$$\Lambda(X) \triangleq \prod_{i=1}^{w} (1 - X x_i)|$$

Et

$$L(X) \triangleq \sum_{i=1}^{w} x_i z_i \prod_{j=1, j\neq 1}^{w} (1 - X x_j)$$

On remarque que les racines du polynôme localisateur Λ (X) sont particulières, toutes sont des puissances deα. Si j $\in$ {1,…., w},$\alpha^{-i_{j-1}}$ est une racine de Λ (X), ceci signifié qu'il y a une erreur à la position i_j . Les racines de ce polynôme, appelé polynôme localisateur, permettent de connaître la position des erreurs. De plus, si on évalue le polynôme évaluateur L(X) en $\alpha^{i_{j-1}}$cela nous donne

$$\prod_{i\in Supp(e)\{i_j\}}(\alpha^{i_{j-1}} - \alpha^{i-1}\ i)e_{i_j},$$

On est ainsi capable de connaître la valeur de l'erreur en cette position.

Dans un premier temps, on propose d'exhiber une relation entre le syndrome et ces deux polynômes. Pour ce faire, on propose de diviser le polynôme évaluateur L(X) par le polynôme localisateur Λ(X).

$$\begin{aligned} L(X) / \Lambda(X) &= \sum_{i=1}^{w} x_i z_i \prod_{j=1, j\neq i}^{w}(1 - Xx_i) \setminus \prod_{i=1}^{w}(1 - Xx_i), \\ &= \sum_{i=1}^{w} \frac{x_i z_i}{(1-Xx_i)} \\ &= \sum_{i=1}^{w} x_i z_i \ (\sum_{j=1}^{\infty} Xx_i^{j-1}) \\ &= \sum_{j=1}^{\infty} X^{j-1} \sum_{i=1}^{w}(z_i x_i^{j}) \\ &= \sum_{j=1}^{\infty} X^{j-1}\ e\vartheta_{\alpha^j}\ (e) \\ &\triangleq S_{\infty}(X) \end{aligned}$$

On appelle S_∞(X) la série syndrome de l'erreur e. On sait que $\forall_j \in$ {1,…., δ- 1 , $e\vartheta_{\alpha^j}$(c)=0. et donc $e\vartheta_{\alpha^j}$(y)= $e\vartheta_{\alpha^j}$(e).

Le destinataire est capable de calculer les δ-1 premiers termes de S_∞(X).

De plus, la dernière égalité devient modulo X^δ :

$$L(X) \equiv L(X)\ S_{\delta}(X) \bmod X^{\delta}.$$

Où $S_{\delta}(X) \triangleq S_\infty(X) \bmod X^\delta$et est de degré au plus δ-1. Cette équation est appelée équation clé.

6.5 Les BCH binaires

On s'intéresse ici aux BCH binaires. La notion de code A-couvert fait référence à un couple: code et algorithme de décodage. On commence par présenter l'algorithme de décodage en liste en temps polynomial de Wu pour décoder les BCH binaires.

6.7.1 Wu : un algorithme de décodage en liste en temps polynomial :

Cette méthode décode jusqu' à la borne de Johnson binaire et s'exécute en temps polynomial, c'est un algorithme de décodage en liste en temps polynomial.

6.7.2 Les codes de Reed-Muller du premier ordre poinçonnés

On peut voir les codes de Reed-Muller du premier ordre poinçonnés comme des codes BCH binaires, ce qui a pour avantage de pouvoir appliquer l'algorithme de Wu à ces codes.

Les codes de Reed-Muller du premier ordre poinçonnés sont des $[2^m - 1, m + 1, 2^{m-1} - 1]_2$-code linéaires pour $m > 1$[14].

7 Conclusion

Dans ce chapitre, nous avons présenté des concepts mathématiques sur les codes correcteurs. Nous avons proposé les codes correcteurs d'erreurs, les codes et distance de Hamming, les codes linéaires et les codes BCH.

CHAPITRE 3

Algorithme de Chiffrement basé sur les codes de correcteurs d'erreurs

1 INTRODUCTION

Dans ce chapitre nous présentons les Algorithmes de Chiffrement basés sur les codes de correcteurs d'erreurs McEliece, Niederreiter.

Nous commençons par des Cryptosystèmes basés sur des codes correcteurs. Ensuite nous proposons le cryptosystéme de Mceliece, ses avantages, ses inconvénients et le cryptanalyse de Mceliece. Enfin nous passerons au principe général de niederreiter.

2 Cryptosystèmes basés sur des codes

– introduits au même moment que RSA par McEliece.
– intérêts
  – plus rapides.
  – meilleures attaques sont exponentielles.
  – basés sur des problèmes difficiles (décodage par syndrome ...).
  – non basés sur la théorie des nombres.
  – alternatifs.
  – ne demandent pas de crypto-processeur.
  – résisteraient à l'ordinateur quantique.
– inconvénient
  – très grande taille des clés publiques (plusieurs centaines de milliers de bits)

3 McEliece

Le premier, McEliece eut l'idée, en 1978, d'utiliser la théorie des codes correcteurs d'erreurs à des fins cryptographiques, et plus précisément pour un algorithme de chiffrement asymétrique.

Le principe du protocole qu'il décrivit consiste à faire envoyer par Alice un message contenant un grand nombre d'erreurs, erreurs que seul Bob sait détecter et corriger [18].

3.1 Cryptosystème de McEliece

C'est le plus ancien crypto système à clef publique utilisant des codes correcteurs d'erreurs. Il a été imaginé par McEliece en 1978, à peu près en même temps que RSA. Comme tous les crypto systèmes à clef publique, ce système est constitué de 3 algorithmes :

1. la génération de clefs.
2. le chiffrement (utilisant la clef publique).
3. le déchiffrement (utilisant la clef secrète).

McEliece a suggéré d'utiliser les codes de Goppa, qui sont des codes linéaires avec un algorithme rapide de décodage. On se propose de le faire avec le code cyclique de Hamming [16].

3.1.1 Cryptosystème de McEliece (idée de base)

- Générer un code que l'on sait décoder et sa matrice génératrice G.

  ceci est la clé privée.
- Transformer G pour obtenir G' qui semble aléatoire.

  ceci est la clé publique.
- chiffrer un message *m* en calculant : $c' = m\,G' \oplus e$

  avec *e* une erreur aléatoire de poids la capacité de correction du code.

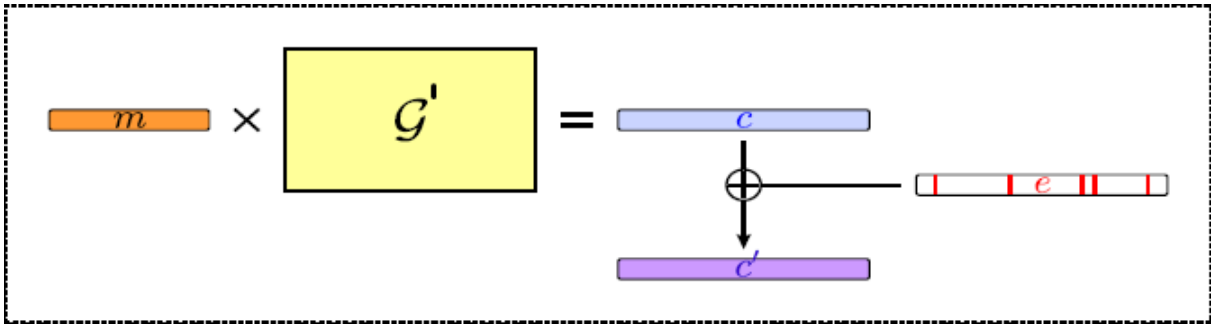


Figure3.1 Cryptosystème de McEliece

Données privées

G : matrice génératrice d'un code C que l'on sait décoder

P : matrice de permutation de taille $n \times n$

Q : matrice inversible de taille $k \times k$

γ_G : algorithme de décodage jusqu'à $\frac{d}{2}$ erreurs

Données publiques

G' $\stackrel{def}{=}$ *QGP*

t : entier < $\frac{d}{2}$ [18]

En réalité, la clef publique du système n'est autre qu'une matrice génératrice d'un autre code linéaire, *C'* équivalent au code *C* — ce nouveau code est donc également un code de distance minimale *d*. En effet, effectuer le produit *G' = SGP* revient simplement à modifier la base choisie pour représenter le code sous forme d'une matrice génératrice (combinaison linéaire des lignes par la matrice *S*) et à permuter les coordonnées du code (permutations des colonnes par la matrice *P*). Je désignerai par la suite ce code *C'* par le terme *code public*, par opposition au *code secret*, *C*. Un message chiffré par le système, *c* s'écrit donc sous la forme d'un mot du code public dont *t* positions sont erronées. Retrouver le texte clair, ou de façon équivalente le vecteur d'erreurs *e*, revient alors à décoder *c* relativement au code public jusqu'à la distance *t*, *i.e.* résoudre, pour *w = t*, le problème suivant :
Problème du décodage jusqu'à la distance w :

Entrée

– G: matrice binaire k × n, de rang k.

– x : vecteur binaire de longueur n

Problème

Trouver, s'il existe, un vecteur binaire m de longueur k tel que d(x,mG) ≤ w[15]

3.1.2 Chiffrement

Soit *m* le message clair que l'on veut envoyer

1. choisir *e* une erreur aléatoire de poids t.

2. calculer $c' = mG' \oplus e$;

• *c'* est le texte chiffré [18].

Dans la variante de McEliece, le chiffrement consiste à multiplier un vecteur de $\mathbb{F}_2^k$ par une matrice binaire $k \times (n - k)$, puis à modifier *t* bits choisis au hasard dans le mot de code correspondant. Bien que cela puisse constituer une difficulté dans la mise en œuvre effective, le coût de l'ajout de l'erreur peut être néglige. En nombre d'opérations binaires, la complexité du chiffrement sera de $\approx \frac{1}{2} k (n - k)$ [16].

3.1.3 Déchiffrement

1. calculer $c' P^{-1} = (mQ) G \times PP^{-1} \oplus eP^{-1}$;

• eP^{-1} est de poids t et (mQ) G est un mot du code ;

2. à l'aide de G on décode et on retrouve (mQ) G ;

3. prendre les k premiers bits de (mQ) G, que l'on note $\tilde{m}$.

• on obtient alors $\tilde{m}$ = mQ car G a été prise sous forme systématique.

4. calculer $\tilde{m}Q^{-1}$ = (mQ) Q^{-1} pour obtenir m.

3.2 Paramètres du système

La clé publique G' est de taille kn. La taille de la clé privée est de l'ordre de O(n) (en prenant un ensemble court générant Q et P). La complexité du chiffrement est celle d'un produit matric-evecteur, soit O (kn), la complexité du déchiffrement est celle du décodage du code soit en général de l'ordre de O (n^2).

Si on prend k de l'ordre de $\frac{n}{2}$ ou$\frac{n}{3}$, on obtient alors que la taille de la clé publique (ainsi que le chiffrement et le déchiffrement) est en O (n^2).

En pratique, McEliece a proposé d'utiliser la famille des codes de **Goppa binaires irréductibles** pour des paramètres [1024; 524; 101] avec t = 50.

Ces paramètres, proposés en 1978 pour une complexité de ≈ 2^{65} (opérations binaires), n'ont été cassés pratiquement qu'en 2008.

Ils conduisent à une **taille de clé de l'ordre de 500 000 bits**. Pour obtenir une complexité en 2^{80}, il conviendrait de prendre des codes de Goppa binaires irréductibles de paramètres [2048; 1600; 46].

L'analyse de complexité précédente montre donc que ce système est beaucoup **plus rapide** que R.S.A, pour un même n de l'ordre de 1024 bits, on obtient un déchiffrement cubique (pour R.S.A) contre un déchiffrement quadratique (pour McEliece). Mais la **très grosse taille de clé publique** reste le problème principal pour l'utilisation pratique.

On peut potentiellement utiliser ce système avec de nombreuses familles de codes. Néanmoins, si l'on utilise une famille trop structurée comme les codes de Reed-Solomon généralisés ou les codes de Reed-Muller, il est possible de faire des attaques structurelles qui permettent de retrouver la structure du code masqué et les matrices Q et P.

L'intérêt des codes de Goppa est qu'il s'agit d'une famille très importante de codes qu'on sait décoder et pour lesquels la structure (de par leur grand nombre) est plus difficile à retrouver [18].

3.3 Les avantages de McEliece

Le principal avantage de cette méthode est sa facilité d'implémentation :
les seules opérations sont des opérations bit à bit. Par contre, l'implémentation nécessite beaucoup de place mémoire.

Ce cryptosystème, reposant sur un problème difficile de la théorie des codes, n'a pas rencontré de véritable soutien dans la communauté cryptographique. L'une des principales raisons de cet état de fait est la taille de la clé. Pourtant, le cryptosystème de McEliece possède des propriétés intéressantes, citons notamment

– la sécurité croît beaucoup plus avec la taille des clés que pour le système RSA ;

– la rapidité du chiffrement.

Un autre avantage est de reposer sur un problème très différent des algorithmes asymétriques usuels.

En conséquence de quoi une percée théorique dans le domaine de la factorisation, qui ruinerait RSA, n'affecterait en rien ce cryptosystème. Le cryptosystème de McEliece résiste à ce jour à toute tentative de cryptanalyse, mais est rarement utilisé en pratique du fait de la grande taille des clés [17].

3.4 Les inconvénients de McEliece

1. La taille de la clef publique (G') est grande. Ceci posera certainement des problèmes d'exécution.
2. Le message chiffré est plus long que le message clair. Cette augmentation de la largeur du message chiffré rend le système plus sensible aux erreurs de transmission.
3. Le crypto système n'est employé pour la signature ou l'authentification parce que l'algorithme de chiffrage n'est pas linéaire et tout l'algorithme est vraiment asymétrique [15].

3.5 Cryptanalyse du système de McEliece

La sécurité repose sur l'impossibilité de déduire un algorithme de décodage efficace de la seule clé publique.

Si tel est le cas, le décryptage d'une instance du système se ramènera à la résolution d'une instance d'un problème de décodage dans un code linéaire. Il en découle que toute attaque du système devra appartenir à l'une des catégories suivantes :

– les attaques par décodage : il s'agit de décoder une instance du cryptosystéme à l'aide d'un algorithme général (i.e. le code public est considère comme un code linéaire aléatoire).

– les attaques structurelles : il s'agit, à l'aide de la clé publique, de retrouver tout ou partie de la structure du code secret [16].

4 Niederreiter

C'est une variante de McEliece reposant sur le problème de décodage par syndrome. L'idée consiste à mettre l'information dans l'erreur et à envoyer un syndrome de cette erreur.

Dans la variante de Niederreiter nous devons calculer le syndrome correspondant à un mot de poids t c'est-à-dire faire la somme de t colonnes de H.

En moyenne, seulement Rt de ces colonnes seront effectivement additionnées, les autres seront dans la partie ((identité)) de la matrice H et auront pour seul effet de modifier un bit dans le résultat, soit environ Rt (n − k) + (1 − R) t ≈ Rt (n − k) opérations binaires [16].

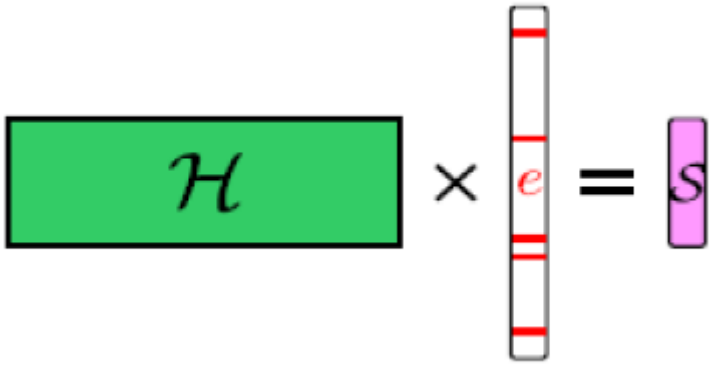


Figure3.2 Décodage par Syndrome

Données privées

H : matrice de parité d'un code C $[n, k, d]$ sur $\mathbb{F}_q$

P : matrice de permutation de taille $n \times n$

Q : matrice inversible de taille $(n - k) \times (n - k)$

y_H : Algorithme de décodage jusqu'à $\frac{d}{2}$ erreurs

Données publiques

$H' \stackrel{def}{=} QHP$

t : entier $< \frac{d}{2}$

4.1 Chiffrement

Convertir les données en erreur *e* de longueur *n* et de poids *t* ;

Calculer $S = H'^{t}e$ (somme de *t* colonnes de H') ;

S est le texte chiffré.

4.2 Déchiffrement

Calculer $Q^{-1}S = Q^{-1}QH(Pe)$;

Pe est de poids *t* et peut être décodé ;

Calculer $P^{-1}(Pe)$ pour obtenir *e* ;

Reconvertir *e* en texte clair [18].

4.3 Principe général du système de Niederreiter

Le système proposé par Niederreiter correspond à une approche duale du système de McEliece. Cette fois-ci, la clef publique est une matrice de parité du code public *C'*.

Le texte clair correspond à un vecteur de longueur *n* et de poids *t*, et le texte chiffré à son syndrome relativement au code *C'*. Retrouver le texte clair à partir du chiffré revient donc ici à déterminer l'unique mot de poids *t* ayant pour syndrome *c*, *i.e.* résoudre, pour *w* = *t*, le problème suivant :

4.3.1 Problème de la recherche des mots de poids inférieur à *w* d'un coset

Entrée

– H : matrice binaire n × r, de rang r.

– s : vecteur binaire de longueur r.

Problème

Trouver, s'il existe, un vecteur binaire e de longueur n et de poids inférieur ou égal à w tel que : eH = s.

4.4 Paramètre

Niederreiter proposait d'utiliser ce système soit avec un code binaire [104, 24,32] résultant de la concaténation du code de Hemming étendu de longueur 8 et dimension 4 avec un code de Reed-Solomon poinçonné de longueur 13 et de dimension 6 défini sur $\mathbb{F}_{16}$, soit avec un code de Reed-Solomon [30,12,19] sur $\mathbb{F}_{31}$. Les paramètres de ces codes sont évidemment trop

petits pour que le système résiste à la cryptanalyse, comme l'ont montré Brickell et Odlyzko en utilisant l'algorithme LLL. La comparaison que j'établis entre les systèmes de McEliece et de Niederreiter, qui n'a donc de sens que si les familles de codes utilisées dans les deux cas sont identiques. Même si leurs sécurités sont équivalentes, ces deux systèmes présentent en effet des différences notables.

L'utilisation du système de Niederreiter permet de réduire de moitié la taille des clefs publiques. Elle autorise en effet le stockage de la matrice de parité *H'* sous forme systématique, qui était impossible pour le système de McEliece puisqu'il aurait révélé une partie du texte clair [15].

Un autre avantage du système de Niederreiter, contrairement au système de McEliece, il produit un chiffrement complétement déterministe puisque les différents chiffrés d'un même texte clair sont toujours identiques. On détecte donc immédiatement que deux chiffrés correspondent au même texte clair mais on ne dispose pas d'algorithme rapide pour les déchiffrer.

5 Conclusion

Dans ce chapitre, nous avons présenté les Algorithme de Chiffrement basé sur les codes de correcteurs d'erreurs. Nous avons proposé les cryptosystèmes Mceliece et ses avantages et ses inconvénients et le cryptanalyse de Mceliece. Enfin nous avons passé au principe général de niederreiter.

CHAPITRE 4

Schémas d'identification

1 INTRODUCTION

Dans ce chapitre, nous commencerons par la définition de preuve de connaissance à divulgation nulle et l'identification Zero-Knowledge ; ensuite nous passerons au schéma d'identification Stern et puis la base de ce schéma et son protocole ensuite leur algorithme .Ce chapitre comporte aussi le Schéma d'identification Veron dont le premier point est la description de ce schéma et le deuxième est son algorithme. Nous proposerons le dernier schéma Cayrel et son protocole et enfin nous passerons aux types de matrice : Random, Quasi-cyclique et Quasi-diadique.

2 Preuve de connaissance à divulgation nulle

Protocole cryptographique permettant à un fournisseur de preuve de convaincre un vérificateur de la validité d'un énoncé (pour lequel il détient une preuve) mais sans révéler aucune autre information que la véracité de cet énoncé [20].

Dans quel contexte utiliser le zero-knowledge ?

- Le prouveur prétend connaitre un secret
- Il veut convaincre le vérificateur qu'il connait un tel secret
- Il veut le faire sans transférer cette connaissance au vérificateur
- Le vérificateur veut être sûr qu'il a en face de lui quelqu'un qui connait effectivement Le secret adéquat.

ZK = Divulgation nulle de connaissance

Ces preuves sont interactives randomisées

2.1 Identification Zero-Knowledge

Zero-knowledge = à divulgation nulle de connaissance (ou encore: sans fuit d'information)

But: Prouveur que l'on a la connaissance d'un secret sans transférer cette connaissance.

- Le vérificateur (celui que l'on veut convaincre) ne doit pas être capable de rejouer le protocole en se substituant au prouveur
- l'interactivité est essentielle, le non-déterminisme également [7]

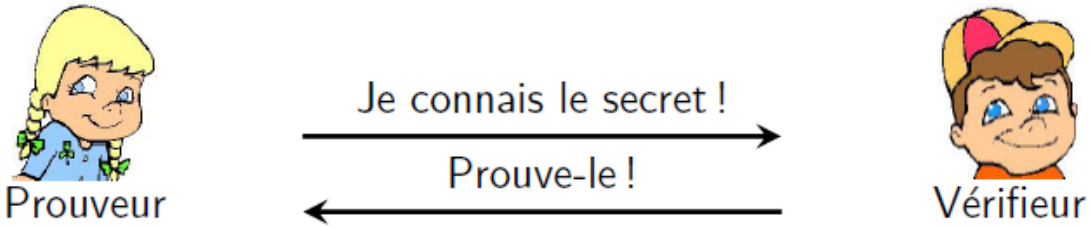


Figure4.1 Identification Zero-Knowledge

3 Schéma d'identification de Stern

Il s'agit d'un système d'identification **à divulgation nulle de connaissance**, basé sur le problème du décodage par syndrome, pour chaque tour : probabilité de triche de $\frac{2}{3}$. Pour une sécurité de 280, il faut 150 tours La norme ISO/IEC-9798-5 préconise deux probabilités : 2^{-16} et 2^{-32}, soit 28 et 56 tours La clé publique est d'environ 200 000 bits. [16].

Le schéma Stern est un protocole d'identification à multiples itérations où chaque itération est une interaction à trois passages entre le prouveur et le vérificateur.

Il utilise un r×n matrice de contrôle de parité H sur le champ binaire $\mathbb{F}_2$, un poids w et une fonction de hachage h commun à tous les utilisateurs.

Chaque prouveur génère ou reçoit une clé secrète s_p $\in \mathbb{F}_2^n$ de poids de Hamming w et calcule un identifiant public $I_p \in \mathbb{F}_2^r$ comme $I_p = H \cdot s_p^T$.

Un utilisateur P est capable de prouver à un vérificateur V pour être celui de posséder la clé secrète s_p pour identifier elle-même. [19].

3.1 Le schéma de base

Le schéma de Stern est un schéma interactif à divulgation nulle de connaissance qui a pour protagonistes un prouveur noté P et un vérifieur noté V. Le prouveur cherchera à s'identifier auprès du vérifieur. Soient n et k deux entiers tels que n ≥ k. Le schéma de Stern nécessite une (n-k) ×n matrice publique H' définie sur$\mathbb{F}_2$. Soit t ≤ n un entier.

La matrice H' et le poids t sont des paramètres du protocole et seront utilisés par plusieurs prouveurs différents. Chaque prouveur P reçoit une clé secrète de n-bits s_p (aussi notée s s'il n'y a pas d'ambiguité sur le prouveur) de poids de Hamming t et calcule un identifiant public

i_P tel que $i_P = H's_P^T$.Cet identifiant est calculé une fois pour toute pour H' donnée et peut alors être utilisé pour de nombreuses authentifications.

Quand un utilisateur P a besoin de prouver à V qu'il est la personne associée à son identifiant public i_P, alors les deux protagonistes suivent le protocole suivant où h est une fonction de hachage standard.

Remarque

Lorsque b vaut 1, durant la quatrième étape du schéma de Stern, on peut noter que $H'y^T$ vient directement de la relation $H'(y \oplus s)^T$ car nous avons :

$H'y^T = H'(y \oplus s)^T \oplus i_P = H'(y \oplus s)^T \oplus H's^T$.

Le protocole est à divulgation nulle de connaissance. Pour un tour d'exécution, la probabilité qu'une personne malhonnête réussisse à tricher est de 2/3.

Afin d'obtenir un degré de sécurité ß, le protocole doit être exécuté un nombre de fois k tel que l'on ait la relation $(2/3)^k \leq ß$. Quand le nombre d'itérations vérifie la dernière condition, alors la sécurité du problème repose sur le problème NP-complet de décodage par syndrome (SD) [17].

3.2 Le protocole de Stern généralisé

1. [Engagement]
   - Chaque signataire choisit aléatoirement $y_i \in \mathbb{F}_2^n$ et une permutation aléatoire σ_i de $\{1,2,\ldots,n\}$ puis envoie à L les engagements $c_{1,i}$, $c_{2,i}$ et $c_{3,i}$ tels que :

     $c_{1,i} = h(\sigma_i \setminus H_i y_i^t)$; $c_{2,i} = h(\sigma_i(y_i))$; $c_{3,i} = h(\sigma_i(y_i \oplus s_i))$

     Où $h(a_1|\ldots|a_j)$ représente le haché de la concaténation de la suite formée par $a_1 \ldots a_j$.
   - L choisit aléatoirement N - t vecteurs $y_i \in \mathbb{F}^n$ et N - t permutations aléatoires σ_i de $\{1,2,\ldots,n\}$.
   - L fixe le secret si des N-t utilisateurs manquants à 0 et calcule les N-t engagements correspondants en choisissant des y_i aléatoires et σ_i $(t + 1 \leq i \leq N)$.
   - L choisit une permutation constante aléatoire de n-blocs Σ sur N blocs $\{1,\ldots,N\}$ dans le but d'obtenir les engagements maîtres :

     $C_1 = h(\Sigma|c_{1,1}|\ldots|c_{1,N})$; $C_2 = h(\Sigma(c_{2,1},\ldots,c_{2,N}))$; $C_3 = h(\Sigma(c_{3,1},\ldots,c_{3,N}))$:
   - L'envoie C_1, C_2 et C_3 à V.

2. [Challenge] V envoie un challenge $b \in \{0, 1, 2\}$ à L qui envoie b aux t signataires.

3. [Réponse] Soit P_i un des t signataires. La première partie de l'étape se déroule entre chaque signataire et L:

Trois possibilités :

- si b = 0 : P_i révèle y_i et σ_i.
- si b = 1 : P_i révèle $(y_i \oplus s_i)$ (noté $(y \oplus s)_i$) et σ_i:
- si b = 2 : P_i révèle σ_i (y_i) (noté $(\sigma(y))_i$) et σ_i (s_i) (noté $(\sigma\ (s))_i$).

L simule les N - t autres protocoles de Stern avec $s_i = 0$ et $t + 1 \leq i \leq N$.

L calcule la réponse pour V (et l'envoie) :

- si b = 0 : L construit $y = (y_{1,\ldots,}\ y_N)$ et $\Pi = \Sigma \circ \sigma$ (avec $\sigma = (\sigma_1,\ldots, \sigma_N)$) et révèle y et Π
- si b = 1 : L construit $y \oplus s = ((y \oplus s)_1 ,\ldots, (y \oplus s)_N)$ et révèle $y \oplus s$ et Π.
- si b = 2 : L construit Π (y) et Π (s) et les révèle.

4. [Vérification] Trois possibilités :

- si b = 0 : V vérifie que Π est une permutation de n-blocs et que C_1,C_2 ont été obtenus de manière honnête.
- si b = 1 : V vérifie que Π est une permutation de n-blocs et que C_1,C_3 ont été obtenus de manière honnête.
- si b = 2 : V vérifie que C_2,C_3 ont été obtenus de manière honnête, et que le poids de Π (s) est tw et que Π (s) est formé de N blocs de longueur n et de poids w ou 0.

5. Réitérer les étapes 1,2,3,4 jusqu'à ce que le niveau de sécurité recherché soit atteint [5].

3.3 Algorithme de schéma d'Identification Stern

Paramètres: n: code length; k: code dimension; H $\in \mathbb{F}_2^{(n-k)\times n}$: parity-check matrix; h a collision resistant hash-function.

Private Key: s $\in \mathbb{F}_2^n$, such that wt(s) = w

Public key: y $\in \mathbb{F}_2^{n-k}$, such that $Hs^T = y$.

- Prover: make commitments

1: Choose u from $\mathbb{F}_n$ at random

2: Choose σ permutation over {1…, n} at random

3: Set $c_1 \leftarrow h((\sigma, Hu^T))$

4: Set $c_2 \leftarrow h(\sigma\ (u))$

5: Set $c_3 \leftarrow h(\sigma\ (u \oplus s))$

6: Send c_i to Verifier

- Verifier: make a challenge

7: Choose challenge b from {0,1,2} at random

8: Send b to Prover

- Prover: answer the challenge

9: **if** b = 0 **then** send u and σ to Verifier

10: **else if** b = 1 **then** send u $\oplus$ s and σ to Verifier

11: **else if** b = 2 **then** send σ (u) and σ (s) to Verifier

12: **end if**

- Verifier: checks the answer complies with commitments

13: **if** b = 0 **then** check if c_1 and c_2 were honestly computed

14: **else if** b = 1 **then** check if c_1 and c_3 are correct.

15: **else if** b = 2 **then** check if c_2 and c_3 are correct, and that wt(σ (s)) = w

16: **end if** [21].

4 Schéma d'identification de Véron

L'idée de Pascal Véron est de reprendre la présentation originale de McEliece et d'exprimer le protocole d'identification sous sa forme duale. En utilisant la matrice génératrice d'un code et non sa matrice de parité [1].

- **Informations publiques partagées par tous les utilisateurs :**

G: matrice binaire k × n aléatoire de rang k.

h : fonction de hachage.

w : entier positif.

- **Clef secrète d'un utilisateur :**

m: vecteur binaire de longueur k.

e : vecteur binaire de longueur n et de poids w.

- **Clef publique d'un utilisateur :**

i = mG + e : vecteur binaire de longueur n.

- **Protocole permettant à Alice de s'identifier auprès de Bob (en τ passes) :**

1. Alice choisit un mot binaire u de longueur k aléatoire et une permutation σ de {1, . . ,n} Elle calcule et envoie à Bob les trois valeurs :

   $c_1 = h(\sigma)$, $c_2 = h(\sigma((u + m)G))$, $c_3 = h(\sigma(uG + i))$

2. Bob choisit un aléa b $\in$ {0, 1, 2} qu'il transmet à Alice.

3. - Si b = 0, Alice révèle u + m et σ à Bob, qui vérifie que c1 = h(σ) et c2 = h(σ ((u + m)G)).

- Si b = 1, Alice révèle σ ((u + m) G) et σe à Bob, qui vérifie que c_2 = h (σ ((u + m) G)), que c_3 = h (σ ((u + m) G) + σe), et que le poids de σe est bien w.
- Si b = 2, Alice révèle σ et u à Bob, qui vérifie que c_1 = h (σ) et c3 = h (σ (uG + i)).

La clef publique de l'utilisateur peut être stockée sur nk + n bits. Chacune des τ passes du protocole nécessite en moyenne $\frac{8}{3}$ k (n − k) + n + $\frac{5}{3}$ k Operations binaires et la transmission de 3h + $\frac{2}{3}$ (k + ℓ_σ + n) bits.

4.1 Algorithme de Schéma d'Identification Véron

Paramètres: n: code length; k: code dimension; G Є $\mathbb{F}_2^{k\times n}$: generator matrix, h a collision resistant hash function.

Private key: (m, e) Є $\mathbb{F}_2^k \times \mathbb{F}_2^n$, such that wt (e) = w

Public key: x Є $\mathbb{F}_2^n$, such that mG + e = x

- Prover: make commitments

1: Choose u from $\mathbb{F}_2^k$ at random

2: Choose σ permutation over {1,…, k} at random

3: Set c1← h (σ)

4: Set c2 ← h (σ ((u + m) G))

5: Set c3 ← h (σ (uG + x))

6: Send c_i to Verifier, i =1, 2, 3

- Verifier: make a challenge

7: Choose challenge b from {0, 1, 2} at random

8: Send b to Prover

- Prover: answer the challenge

9: if b = 0 then send u + m and σ to Verifier

10: else if b = 1 then send σ ((u + m) G) and σ (e) to Verifier

11: else if b = 2 then send σ and u to Verifier

12: end if

- Verifier: checks the answer complies with commitments

13: if b = 0 then check if c1 and c2 were honestly computed

14: else if b = 1 then check if c2 and c3 are correct, and wt (σ (e)) = w

15: else if b = 2 then check if c1 and c3 are correct.

16: end if [21]

5 Schéma d'identification de Cayrel-Veron-El Yousfi

Considérons un élément de $\mathbb{F}_q^n$ comme n blocs de taille $[\log_2(q)] = N$.

On représente chaque élément de $\mathbb{F}_q$ par N bits.

Définition

Soit une permutation de l'ensemble $\{1, \ldots, n\}$ et $\gamma = (\gamma_1, \ldots, \gamma_n) \in \mathbb{F}_q^n$ Telle que $\forall_i, \gamma_i \neq 0$.

On définit la transformation $\Pi_{\gamma,\Sigma}$ par :

$$\Pi\Pi\gamma, \Sigma : \mathbb{F}_q^n \rightarrow : \mathbb{F}_q^n$$

$$\vartheta \rightarrow (\gamma\Sigma_{(1)}\, \vartheta\Sigma_{(1)}, \ldots.., \gamma\Sigma_{(n)}\, \vartheta\Sigma_{(n)})$$

On remarque que $\forall \alpha \in \mathbb{F}_q$, $\forall\, \vartheta \in \mathbb{F}_q^n$: $\Pi_{\gamma,\Sigma}(\alpha\vartheta) = \alpha\, \Pi_{\gamma,\Sigma}(\vartheta)$ et $wt(\Pi_{\gamma,\Sigma}(\vartheta)) = wt(\vartheta)$.

Génération de clefs

Soit $r = n-k$. Le schéma utilise une $(r \times n)$ matrice q-aire aléatoire H commune à tous les utilisateurs.

Elle peut être considérée comme la matrice de parité d'un **code** (n, k) **linéaire aléatoire** sur $\mathbb{F}_q$. Sans perte de généralité, on peut supposer que H est donnée sous la forme $H = (I_r|M)$ où M est une matrice $r \times (n-r)$ aléatoire.

Soit k un paramètre de sécurité.

On choisit n, r, t et q tels que $WF_{ISD}(n, r, t, q) \geq 2^k$.

On choisit aléatoirement une matrice H (q-aire de taille $(r \times n)$).

On prend aléatoirement un élément $e \in \mathbb{F}_q^n$ avec $wt(e) = t$.

On calcule S tel que $H^t e = S$.

On a pk = (S, H, t) et sk = e.

La démarche est la même que pour le schéma de Stern, sauf que l'on se place dans $\mathbb{F}_q$.

Identification

Le détenteur du secret e **va prouver au vérificateur qu'il le connaît bien** en utilisant deux techniques de masquage : la transformation décrite précédemment et un vecteur aléatoire.

Cependant, un prouveur malhonnête pourra tricher avec une probabilité $\approx 1/2$.

C'est pourquoi, on devra répéter le protocole plusieurs fois, pour atteindre le degré de sécurité désiré.

La sécurité est basée sur la difficulté du problème de décodage par syndrome q-aire.

5.1 Le protocole de Cayrel-Véron-ElYousfi

Prouveur P		**Vérifieur V**
(sk, pk) = (e, (S, H, t))		pk = (S,H, t)
$u \xleftarrow{\$} \mathbb{F}_q^n, \Sigma \xleftarrow{\$} S_n, \gamma \xleftarrow{\$} \mathbb{F}_q^{n*}$		
$c_1 \leftarrow h(\Sigma \Vert \gamma \Vert H^t u)$		
$c_2 \leftarrow h(\Pi_{\gamma,\Sigma}(u) \Vert (\Pi_{\gamma,\Sigma}(e)) \xrightarrow{c_1,c_2}$		
	$\xleftarrow{\alpha} \alpha \xleftarrow{\$} \mathbb{F}_q^*$	
$\beta \leftarrow \Pi_{\gamma,\Sigma}(u + \alpha e)$	$\xrightarrow{\beta}$	
	$\xleftarrow{\text{Challenge b}}$	$b \xleftarrow{\$} \{0, 1\}$
Si $b = 0$:	$\xrightarrow{\gamma,\Sigma}$	**Vérifie** $c_1 \stackrel{?}{=} h(\Sigma \Vert \gamma \Vert H^t \Pi_{\gamma,\Sigma}^{-1}(\beta) - \alpha S)$
Sinon :	$\xrightarrow{\Pi_{\gamma,\Sigma}}$	**Vérifie** $c_2 \stackrel{?}{=} h(\beta - \alpha\Pi_{\gamma,\Sigma}(e) \Vert \Pi_{\gamma,\Sigma}(e))$, $wt(\Pi_{\gamma,\Sigma}(e)) \stackrel{?}{=} t$ [17]

6 Autour de l'identification

6.1 Matrice Random

Considérons la matrice binaire A de taille r×n qui se compose de r lignes et de n colonnes. Le moyen le plus rapide pour calculer la matrice-vecteur est le produit $A \cdot x^T$ pour un vecteur x de longueur n est égal à XOR.

Les colonnes à la position i si l'entrée de i-ième x est égal à 1.

Notre implémentation stocke la matrice A comme un tableau de vecteurs de colonnes, stocke r Bit Séquencées de longueur n.

La structure de données est en mesure d'effectuer une opération XOR avec deux BitSequences nativement le produit matrice-vecteur est assez rapide.

6.2 Matrice Quasi-cyclique

La matrice quasi-cyclique –vecteur produit fonctionne sur les indices de la première ligne de la matrice et calcule le résultat de bit.

Notre implémentation suit cette approche et calcule le produit matrice-vecteur de bits de

cette façon. Mais nous avons réalisé que les opérations sur les bits individuels dans nos BitSequences sont très coûteuses et ralentissent le produit matrice-vecteur.

6.3 Matrice Quasi-diadique

Le matrice quasi-dyadique-vecteur produit calcule le résultat au niveau du bit. Notre implémentation suit cette approche et calcule le produit matrice-vecteur de bits de cette façon. Mais, comme mentionné ci-dessus sur bits individuels ces opérations sont très coûteuses dans notre BitSequence.[19]

7 Conclusion

Dans ce chapitre nous avons présenté la preuve de connaissance à divulgation nulle et les trois schémas d'identification successivement Stern, Veron et Cayrel-Veron-Al Yousfi. et pour chaque schéma leur protocole en détail ; ensuite nous sommes passés aux trois types de matrice Random, Quasi-cyclique et Quasi-diadique.

CHAPITRE 5

Implémentation et test de performance

1 INTRODUCTION

Dans ce chapitre, nous avons implémenté les systèmes de chiffrement à clé publique, déjà vu dans le chapitre précédent. Nous avons testé les performances selon le temps d'exécution et la capacité de correction de chaque algorithme, puis nous avons testé les performances selon le temps d'exécution pour chaque schéma d'identification. Enfin, nous avons comparé nos résultats.

2 Environnement de développement

2.2 NetBeans

NetBeans est un environnement de développement intégré (IDE) pour Java, placé en open source par Sun en juin 2000 sous licence CDDL (Common Development and Distribution License). En plus de Java, NetBeans permet également de supporter différents autres langages, comme Python, C, C++, XML et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, refactoring, éditeur graphique d'interfaces et de pages web).

NetBeans est disponible sous Windows, Linux, Solaris (sur x86 et SPARC), Mac OS X et Open VMS.

NetBeans est lui-même développé en Java, ce qui peut le rendre assez lent et gourmand en ressources mémoires.

2.3 JAVA

Java est un langage de programmation à usage général, évolué et orienté objet dont la syntaxe est proche du C. Ses caractéristiques ainsi que la richesse de son écosystème et de sa communauté lui ont permis d'être très largement utilisé pour le développement d'applications de types très disparates. Java est notamment largement utilisée pour le développement d'applications d'entreprises et mobiles [12].

2.3.1 Fonctions et mécanismes de sécurité offerts par la plate-forme Java

Différentes fonctions et mécanismes de sécurité sont présents ou peuvent être mis en œuvre dans Java. Ces mécanismes peuvent être classés en deux catégories :

– les mécanismes natifs qui s'appliquent à toutes les applications Java de manière relativement transparente pour l'utilisateur et le développeur ;

– les mécanismes optionnels qui peuvent être utilisés par les développeurs pour implémenter des fonctions de sécurité au sein des applications Java.

Dans notre travaille, nous nous concentrons sur les mécanismes optionnels.

La confidentialité du code source peut être assurée en partie par des extensions utilisant des techniques d'offuscation ou de chiffrement de classes. De même, certains outils (notamment d'analyse statique) peuvent être utilisés afin de vérifier l'innocuité du code source.

La bibliothèque standard de Java propose également des mécanismes de sécurité qui peuvent être mis en œuvre pour sécuriser une application Java :

– Les mécanismes cryptographiques permettent d'assurer la confidentialité et l'intégrité des données de l'application via les opérations de chiffrement, de signature et d'authentification.

– Les mécanismes d'authentification et de contrôle d'accès permettent de définir des règles de contrôle d'accès pour les différents composants (classes Java) de la plateforme. Ils permettent également d'auditer en partie les accès réalisés par les applications et les bibliothèques Java.

– Le mécanisme de signature des classes et des fichiers d'archives (JAR) permet de garantir l'intégrité de la provenance des classes chargées par la JVM [11]

3 Caractéristiques d'application

Nous parlons ici sur les caractéristiques de notre application comme l'architecture d'application et les interfaces.

3.1 Architecture de notre application

Notre application contienne une interface principale, elle présente les algorithmes Mceliece et Niederreiter et les schémas d'identifications Stern et Veron.

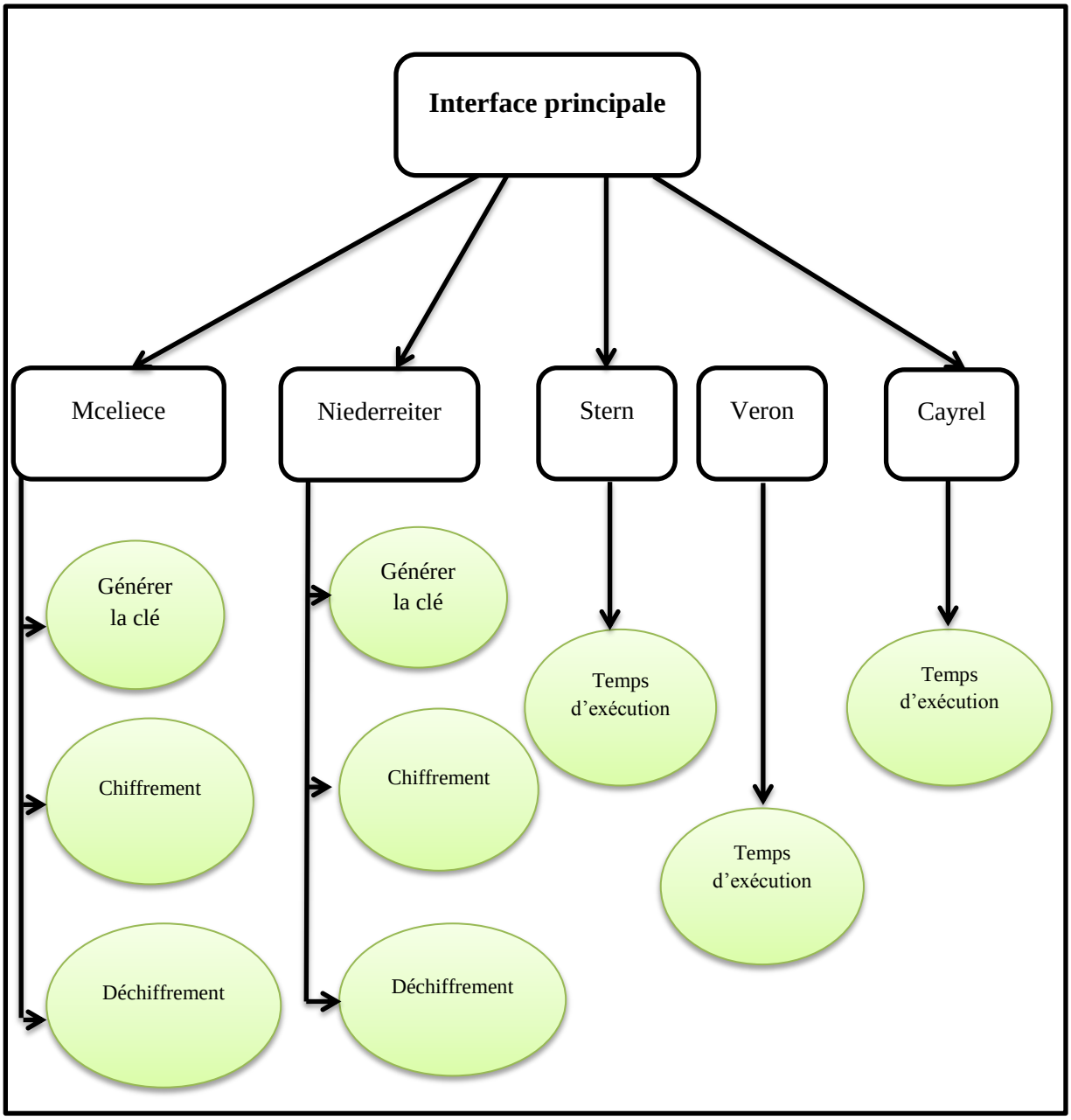
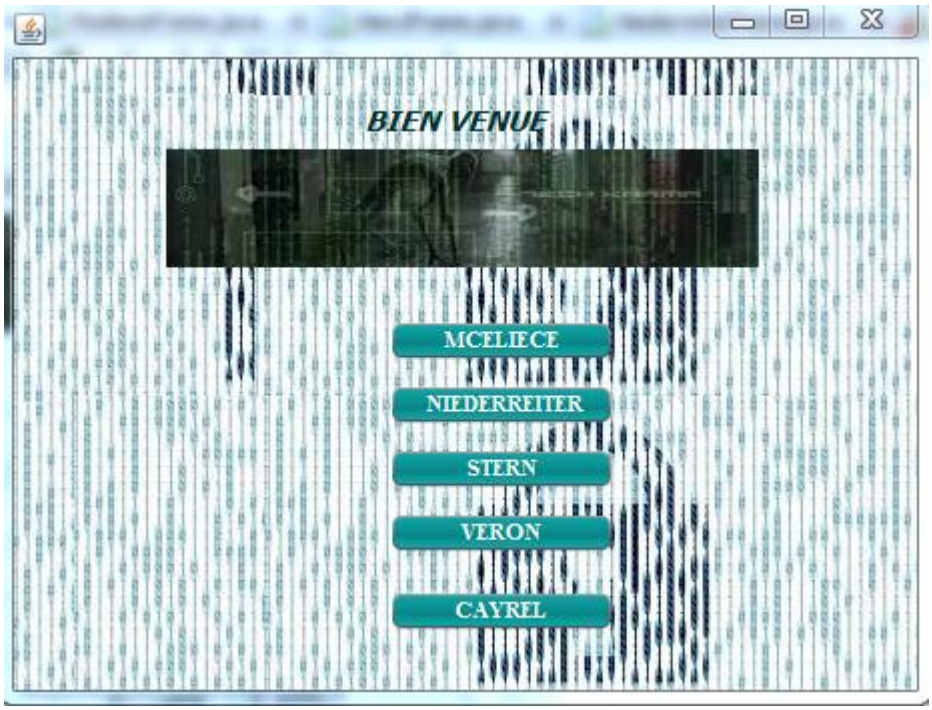


Figure 5.1 Architecture d'application

3.2 Certain interface de notre application

L'interface principale d'application contient des boutons pour accéder aux autres interfaces partiels comme Mceliece, Niederreiter, Stern……..



Si on clique par exemple sur le bouton Stern nous verrons l'interface suivante:

Lorsque je clique sur un bouton de l'interface principale, il me prend à l'interface sélectionné comme Stern dans cette capture, on trouve un bouton pour afficher les résultats de ce schéma d'identification et deux listes la premier pour choisir le nombre d'itération utilisé et

la deuxième pour choisir le type de matrice, si l'on clique sur le bouton ok Nous obtiendrons les résultats suivants :

4 Implémentation et résultats expérimentaux

4.1 Bibliothèques d'extensions relatives à la sécurité

Les API qui paraissent les plus pertinentes en termes d'apport pour la sécurité sont les suivantes :

- API Bouncy Castle Crypto ;
- API FlexiProvider.

Nom	**Editeur**	**Confiance dans l'éditeur**	**Prix**	**Version**
Bouncy Castle Crypto	Groupe	Haute	open source	v1.41 (02/10/2008)
FlexiProvider	Université	Haute	open source	v1.6p3 (2008)

Tableau 5.1 Récapitulatif des API Java de sécurité

Dans notre application, nous utilisons la bibliothèque FlexiProvider pour implémenter les cryptosystemes à clé publique comme Mceliece et Niederreiter et schémas d'identifications Stern et Veron.

4.1.1 FlexiProvider

FlexiProvider est une API de sécurité Java et un provider pour le JCA/JCE de Sun Java développée par le Theoretical Computer Science Research Group dirigé par le professeur Johannes Buchmann de l'université des sciences de Daemstadt (Allemagne).

Elle implémente les algorithmes cryptographiques les plus répandus et dispose en fait de quatre providers différents open source (CoreProvider, ECProvider, PQCProvider, NFProvider).

Le CoreProvider est sous licence LGPL, le ECProvider, PQCProvider et NFProvider sont eux sous licence GPL.

Description des providers

CoreProvider le CoreProvider contient les algorithmes cryptographiques publiques les plus connus, tels que RSA, AES, 3-DES, chiffrement de mot-de-passe selon PKCS#5, MD5, SHA1, RIPEMD, CBC-MAC, CMAC, HMAC et un générateur de nombre pseudo aléatoire (BBS) ;

ECProvider le ECProvider contient les algorithmes cryptographiques dédiés aux courbes elliptiques, tels que ECDSA (Elliptic Curve Digital Signature Algorithm), ECNR, ECDH (Elliptic Curve Diffie-Hellman), ECIES (Elliptic Curve Integrated Encryption Scheme) ;

PQCProvider le PQCProvider contient des algorithmes cryptographiques quantiques expérimentaux, non standardisés, tels que CMSS et CFS (signature), McEliece cryptosystem et quatre de ses variantes, Niederreiter cryptosystem ;

NFProvider le NFProvider est fourni à titre expérimental. Il a été développé pour prouver la viabilité des nombres finis dans un contexte quadratique imaginaire (IQ).

FlexiProvider est compatible avec Sun Java version 1.3.1_18, 1.4.2_12, 1.5.0_10 et 1.6.0_03.

Cette API est disponible à l'adresse suivante :

http://www.cdc.informatik.tu-darmstadt.de/flexiprovider.[11]

4.1.2 Java Crypto API

Fournit les classes et les interfaces pour les opérations cryptographiques. Les opérations cryptographiques définies dans ce package incluent le cryptage, la génération de clés et un accord clé, Message Authentication Code (MAC) génération.

Soutien pour le chiffrement comprend symétrique, asymétrique, bloc, et les chiffrements de flux. Ce paquet prend également en charge les flux sécurisés et objets scellés.

Beaucoup de classes prévues dans ce paquet sont basé sur le fournisseur. La classe se définit une interface de programmation d'applications qui peuvent écrire.

Les implémentations eux-mêmes peuvent alors être écrites par des fournisseurs tiers indépendants et branchés de façon transparente comme nécessaire. Par conséquent les développeurs d'applications peuvent profiter de n'importe quel nombre de mises en œuvre sur la base fournisseurs, sans avoir à ajouter ou réécrire le code [22].

4.2 Calcul du temps d'exécution

Pour chaque application, la fonction de J2SE NanoTime fournit des données de synchronisation. La fonction NanoTime, qui prend aucun argument et renvoie une valeur de type long, réside dans la classe java.lang.System.

Il revient avec précision de la nanoseconde la valeur de la temporisation du système le plus précis disponible;

En utilisant les valeurs retournées par NanoTime avant et après une ou plusieurs opérations, on peut prendre la différence pour obtenir le temps écoulé.

```
public static long currentTimeMillis ()
```

Pour calculer le temps d'exécution il faut utiliser le code suivant :

```
Long startTime, allTime, duration ;
startTime = System.nanoTime();
allTime = System.nanoTime();
duration = allTime – startTime ;
```

4.3 Résultats expérimentaux

4.3.1 Le temps d'exécution

Le temps d'exécution de cryptosystéme Mceliece est représenté par le tableau suivant :

Longueur de code (bits)	64	256	1024	2048
Génération de clé (ms)	100	170	4240	18846

Tableau 5.2 Le temps d'exécution de génération de clé de Mceliece (ms)

Le temps d'exécution de cryptosystéme Niederreiter est représenté par le tableau suivant :

Longueur de code (bits)	64	256	1024	2048
Génération de clé (ms)	10	80	1490	12302

Tableau 5.3 Le temps d'exécution de génération de clé de Niederreiter (ms)

Le tableau ci-dessous présente les résultats de temps d'exécution avec les trois types de matrice, en fonction du paramètre de sécurité et en utilisant une longueur de hachage de 160 bits. Les résultats sont présentés pour un tour et une identification complète en 28 et 56 tours pour Stern.

Paramètres	n = 1024, r = 512, w = 128, rr = 256		
Type de matrice	1 itération	28 itérations	56 itérations
Random (ms) Quasi-cyclique (ms) Quasi-diadique (ms)	251.19 361.13 405.51	6467.39 8217.12 14886.66	12734.26 16014.96 29611.65

Tableau 5.4 Le temps d'exécution de schéma d'identification Stern (ms)

Le tableau ci-dessous présente les résultats de temps d'exécution avec les trois types de matrice, en fonction du paramètre de sécurité et en utilisant une longueur de hachage de 160 bits. Les résultats sont présentés pour un tour et une identification complète en 28 et 56 tours pour Veron

Paramètres	n = 1024, r = 512, w = 128, rr = 256		
Type de matrice	1 itération	28 itérations	56 itérations
Random (ms) Quasi-cyclique (ms) Quasi-diadique (ms)	333.51 346.30 406.98	7078.23 8720.89 14516.12	13261.76 17350.96 29944.66

Tableau 5.5 Le temps d'exécution de schéma d'identification Veron (ms)

Le tableau ci-dessous présente les résultats de temps d'exécution avec les trois types de matrice, en fonction du paramètre de sécurité et en utilisant une longueur de hachage de 160 bits. Les résultats sont présentés pour un tour et une identification complète en 16 tours pour Cayrel-Véron-ElYousfi (pour le même niveau de sécurité).

Paramètres	n = 208, r = 104, w = 78, q = 256 n_prng = 1024, r_prng = 512, w_prng = 128, rr_prng = 256	
Type de matrice	1 itération	16 itérations
Random (ms) Quasi-cyclique (ms) Quasi-diadique (ms)	520.58 210.64 345,17	2290.86 2953.12 3120.36

Tableau 5.6 Le temps d'exécution de schéma d'identification Cayrel (ms)

4.3.2 La capacité de correction

La capacité de correction de cryptosystéme Mceliece selon la longueur de code se représenté par le tableau suivant :

La longueur de Code (n) (bits)	64(bits)	256(bits)	1024(bits)	2048(bits)
Capacité de correction (t) (bits)	5	16	51	93

Tableau 5.7 La capacité de correction de cryptosystéme Mceliece (bits)

La capacité de correction de cryptosystéme Niederreiter selon la longueur de code se représenté par le tableau suivant :

La longueur de Code (n) (bits)	64(bits)	256(bits)	1024(bits)	2048(bits)
Capacité de correction (t) (bits)	5	16	51	93

Tableau 5.8 La capacité de correction de cryptosystéme Niederreiter (bits)

6 Etude Comparative

La comparaison entre les algorithmes Mceliece et Niederreiter sera faite par le calcul du temps d'exécution et la capacité de correction des erreurs pour chaque algorithme. La comparaison entre les schémas d'identification Stern, Veron et Cayrel-Veron-al Yousfi sera faite par le calcul de temps d'exécution.
Tous les résultats sont représentés dans les tableaux suivants.

6.1 Comparaison les différents cryptosystemes

	Mceliece				**Niederreiter**				**Comparaison**			
Longueur de code (bits)	**64**	**256**	**1024**	**2048**	**64**	**256**	**1024**	**2048**	**64**	**256**	**1024**	**2048**
Génération de clé (ms)	100	170	4240	18846	10	80	1490	12302	$\frac{100}{10}$	$\frac{170}{80}$	$\frac{4240}{1490}$	$\frac{18846}{12302}$

Tableau 5.9 Comparaison entre les cryptosystémes Mceliece et Niederreiter selon le temps d'exécution (ms)

Les résultats de ce tableau montrent que la génération de clé de cryptosystème Niederreiter plus rapide que Mceliece en fonction de la longueur du code.

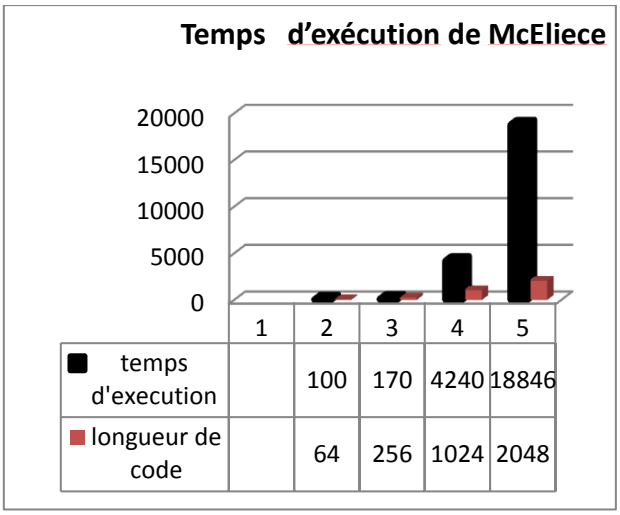


Diagramme1.Le temps d'exécution de génération de clé de Mceliece (ms)

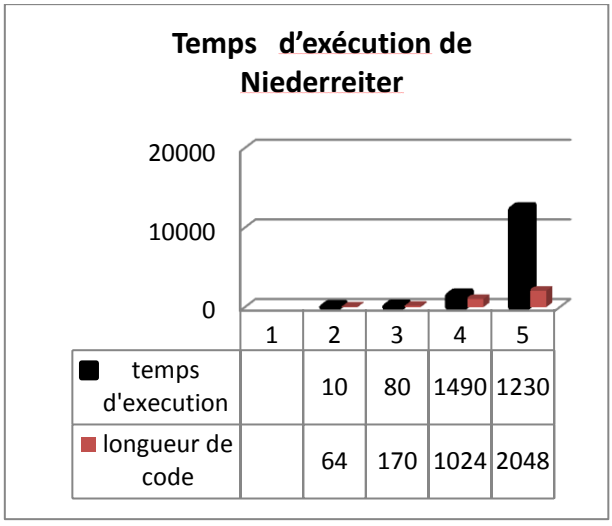


Diagramme2.Le temps d'exécution de génération de clé de Niederreiter (ms)

Les résultats de ces diagrammes montrent que la génération de clé de cryptosystème Niederreiter plus rapide que MceEiece en fonction de la longueur du code.

	Mceliece				**Niederreiter**				**Comparaison**			
Longueur de code(bits)	**64**	**256**	**1024**	**2048**	**64**	**256**	**1024**	**2048**	**64**	**256**	**1024**	**2048**
Capacité de Correction(bits)	5	16	51	93	5	16	51	93	$\frac{5}{5}$	$\frac{16}{16}$	$\frac{51}{51}$	$\frac{93}{93}$

Tableau 5.10 Comparaison entre les cryptosystémes Mceliece et Niederreiter selon la capacité de correction (bits)

Les résultats de ce tableau montrent que la capacité de correction de cryptosystèmes Niederreiter et Mceliece est la même en fonction de la longueur de code.

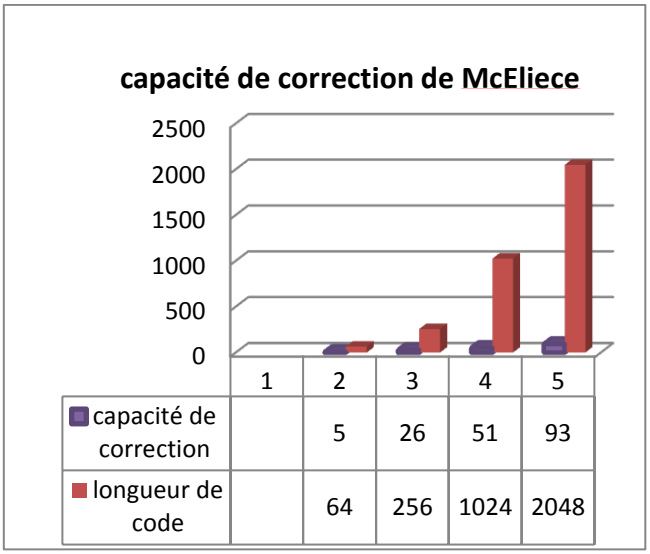


Diagramme3. La capacité de correction de cryptosystéme Mceliece (bits)

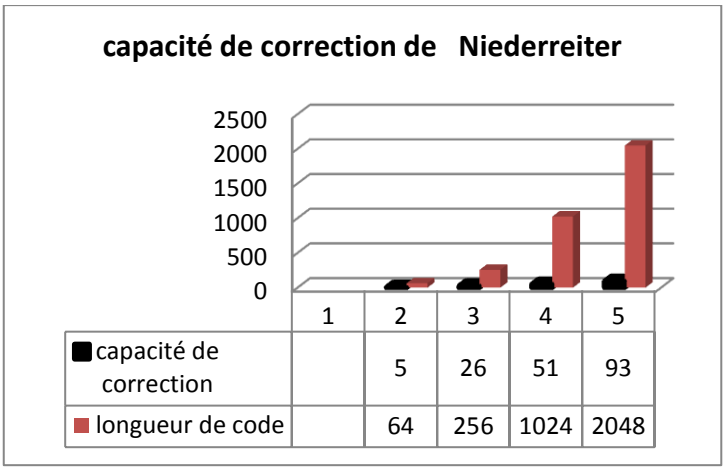


Diagramme4. La capacité de correction de cryptosystéme Niederreiter (bits)

Les résultats de ces diagrammes montrent que la capacité de correction de cryptosystèmes Niederreiter et McEliece est la même en fonction de la longueur de code.

6.2 Comparaison les différents schémas

Les résultats de Comparaison de trois schémas d'identifications Stern, Veron et Cayrel-Veron-El Yousfi sera présentées par le tableau suivant :

	Stern		Veron		Cayrel	
paramètres	n = 1024, r = 512 w = 128, rr = 256		n = 1024, r = 512 w = 128, rr = 256		n = 208, r = 104, w = 78, q = 256 n_prng = 1024, r_prng = 512, w_prng = 128, rr_prng = 256	
Type de matrice	1 itération	28 itérations	1 itération	28 itérations	1 itération	28 itérations
Random(ms) Quasi-Cyclique(ms) Quasi-Diadique(ms)	251.19 361.13 405.51	6467.39 8217.12 14886.66	333.51 346.30 406.98	7078.23 8720.89 14516.12	520.58 210.64 345,17	2290.86 2953.12 3120.36

Tableau5.11 Comparaison des différents schémas (ms)

Pour un niveau de sécurité de 2^{107} respectivement 2^{128}. le schéma de Cayrel et al Yousfi est le plus rapide par rapport à Stern et Veron qui a besoin presque du même temps

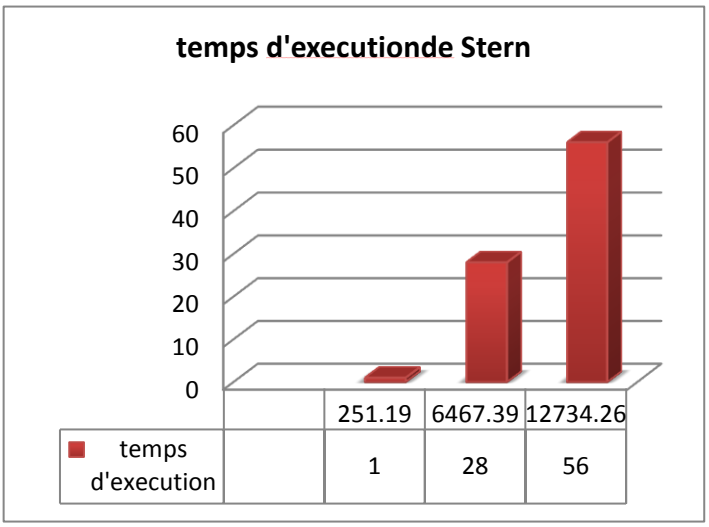


Diagramme5.Le temps d'exécution de Stern (ms)

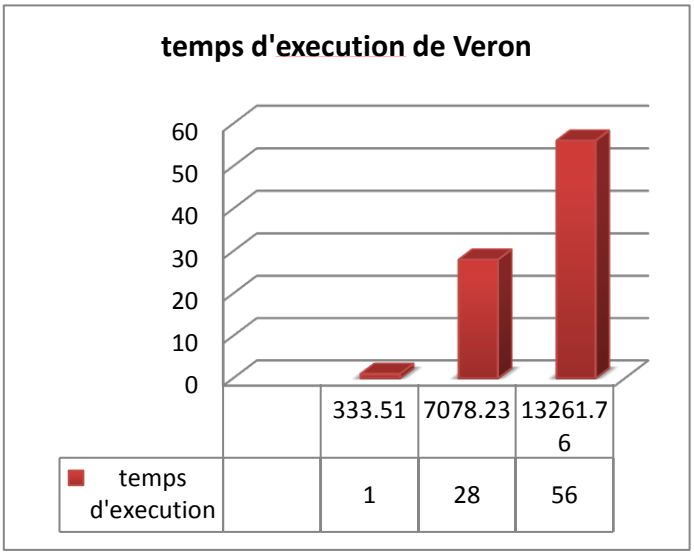


Diagramme6.Le temps d'exécution de Veron (ms)

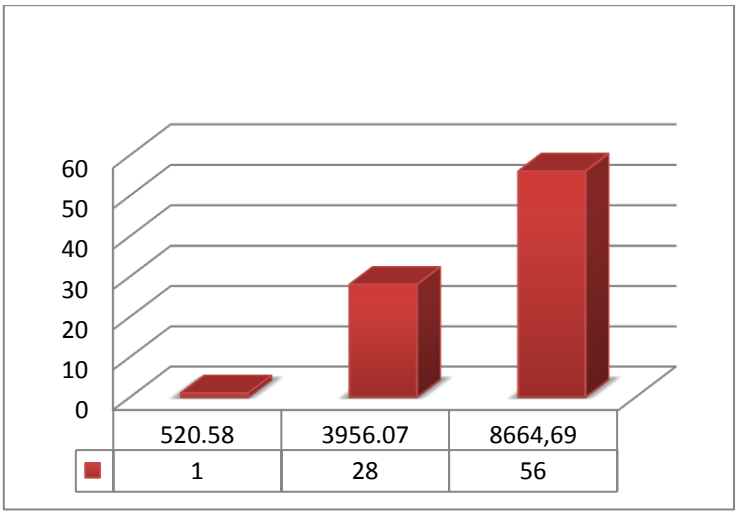


Diagramme7.Le temps d'exécution de Cayrel (ms)

Les résultats de ces diagrammes montrent que le schema cayrel plus rapide que Stern et Veron en fonction de nombres d'itérations.

7 Conclusion

Dans ce chapitre, nous avons présenté l'environnement de développement et le langage de programmation qui a utilisé pour implémenter les systèmes cryptographiques à clé publique que nous avons étudiés dans le chapitre précédente, et nous avons présenté et expliqué leurs fonctionnalités,

Après cela, nous avons testé la performance de différents algorithmes et nous avons comparé les algorithmes, enfin nous avons débattu les résultats obtenus.

Conclusion

Dans ce mémoire nous avons montré comment protéger une information par détection et correction d'erreurs quand la communication se fait dans un canal bruité. C'est pour cette raison que nous utilisons les cryptosystemes à clé publique Mceliece et Niederreiter qui ont une sécurité équivalente, cette sécurité repose sur la difficulté de retrouver la matrice de parité H à partir de H' et le problème du décodage par le Syndrome Pour assurer l'authentification nous utilisons les schémas d'identification Stern, Veron et Cayrel-Veron-El Yousfi qui sont des systèmes d'identification à divulgation nulle de connaissance. Le but de ce système est de montrer que le Prouveur a la connaissance d'un secret sans transférer cette connaissance. Toute cette étude est réalisée pour assurer la sécurité et protéger la vie privée des utilisateurs.

Références bibliographiques

[1] A. CANTEAUT'' Attaques de cryptosystémes à mots de poids faible et construction de fonctions t-résilientes'', Disponible:
https://www.rocq.inria.fr/secret/Anne.Canteaut/Publications/Canteaut-these.pdf
[Consulté le 16/03/2014].

[2] B. Biswas'' Implementational aspects of code-based cryptography'', Disponible: http://pastel.archives-ouvertes.frdocs00523007ANNEXdefence.pdf
[Consulté le 15/04/2014].

[3] '' CHAPITRE 1.INTRODUCTION A LACRYPTOGRAPHIE'',
Disponible: http://math.univ-lyon1.fr/~roblot/masterpro.html [Consulté le 10/01/2014].

[4] C. FAURE, '' Etudes de systèmes cryptographiques construits à l'aide de codes correcteurs, en métrique de Hamming et en métrique rang '', 17 mars 2009. [En ligne].

Disponible: http://www.normalesup.org/~faure/manuscrit.pdf [Consulté le 22/02/2014].

[5] C. Melchor et P. Gaborit '' A new efficient threshold ring signature scheme based on coding theory'', PQ CRYPTO 2008, LNCS, Disponible:

[Consulté le 08/03/2013].

[6] Coste, Paugam et Quarez,'' Le cours Codes correcteurs'', Disponible: http://agreg-maths.univ-rennes1.fr/documentation/docs/codes.pdf.

[Consulté le 5/12/2013].

[7] E, Bresson '' CRYPTOGRAPHIE Identification et Zero Knowledge '', Disponible: http://www.di.ens.fr/~bresson/P12-M1/P12-M1-Crypto_9.pdf

[Consulté le 5/5/2014].

[8] G. Montcouquiol,'' Codes détecteurs et correcteurs d'erreurs'', IUT Orsay, 2006-2007,Disponible:
http://www.google.dz/url?sa=t&rct=j&q=&esrc=s&source=web&cd=10&ved=0CHUQFjAJ&url=http%3A%2F%2Fwww.math.upsud.fr%2F~montcouq%2FEnsei

gnements%2FCodage%2Fcours.pdf&ei=8vEIU_jmGoOH0AXhIDgBw&usg=AFQjCNE0WgEkoPNcElGr57w0eZsbMCTi0g&bvm=bv.61725948,d.Yms.

[Consulté le 5/12/2013].

[9] G.RIBIÈRE, ''Certification électronique et sécurité ''. Disponible: http://www.google.dz/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CCsQFjAA&url=http%3A%2F%2Ftel.archives-ouvertes.fr%2Fdocs%2F00%2F50%2F37%2F71%2FPDF%2Fmanuscrit.pdf&ei=_LQcU9qKNeyf7gae34GAAw&usg=AFQjCNF4WOdC_5HF2qODHUILLFruZk9T3A&bvm=bv.62578216,d.ZGU[Consulté le 22/02/2014].

[10] '' Introduction à la cryptographie'', Master Pro - Ingénierie Mathématique-Cryptographie. Disponible: ftp://ftp.pgpi.org/pub/pgp/6.5/docs/french/IntroToCrypto.pdf. [Consulté le 17/02/2014].

[11] '' JAVASEC_NTE''. Disponible: http://www.ssi.gouv.fr/IMG/pdf/JavaSec-Langage.pdf. [Consulté le 17/04/2014].

[12]J.DOUDOUX '' Développons en JAVA''. Disponible: http://benhur.teluq.ca/SPIP/inf6104/IMG/pdf/dej_0_85.pdf [Consulté le 17/02/2014].

[13] J. vélu, ''méthodes mathématiques pour l'informatique'', DUNOD, Palerme, France, le 27 juin 2005.

[14] M. BARBIER, '' Décodage en liste et application à la sécurité de l'information'', Disponible: https://barbierm01.users.greyc.fr/Documents/thesis/MB_thesis.pdf
[Consulté le 1/04/2013].

[15] N. HADJ-SAID, A.-PACHA, A. BELGORAF , A. M'HAMED ''Crypto Système à Clé Publique de McEliece basé sur les Codes Cycliques de Hamming'', Disponible: http://hal.inria.fr/docs/00/04/38/06/PDF/130.pdf

[Consulté le 30/03/2014].

[16] N. Sendrier'' Cryptosystémes à clé publique basés sur les codes correcteurs d'erreurs'', Institut National de Recherche en Informatique et Automatique, Rocquencourt, Disponible: https://www.rocq.inria.fr/secret/Nicolas.Sendrier/hab.pdf

[Consulté le 16/03/2014].

[17] P. CAYREL'' Construction et optimisation de cryptosystèmes basés sur les codes correcteurs d'erreurs'', Disponible: http://epublications.unilim.fr/theses/2008/cayrel-pierre-louis/cayrel-pierre-louis.pdf
[Consulté le 30/03/2014].

[18] P. CAYREL'' Cours de cryptographie basée sur les codes correcteurs d'erreurs'', Université Jean Monnet, Laboratoire Hubert Curien, Disponible: http://www.cayrel.net . [Consulté le 17/02/2014].

[19] P. CAYREL, S.Al YOUSFI'' Implementation of Code-Based Zero-Knowledge Identification'' Schemes, Disponible: http://www.cayrel.net/IMG/pdf/report_2_.pdf. [Consulté le 17/05/2014].

[20] S. GAMBS, '' Accréditations anonymes'', Disponible: http://www.irisa.fr/prive/sgambs/pvp_cours4.pdf. [Consulté le 17/05/2014].

[21]''Signature'', Disponible : http://docs.oracle.com/javase/7/docs/api/javax/crypto/package-summary.html [Consulté le 2/06/2013].

الملخص

التطور السريع للشبكات العالمية والإمكانيات الهائلة التي توفرها التعاملات الالكترونية في التواصل المستمر, اليوم تشكل مشكلة على حماية المعلومات ضد أخطاء الإرسال من ناحية، ومن ناحية أخرى هذه البيانات غير مفهومة إلا الاشخاص المعنيين.

ترميز المعلومات يستخدم لمكافحة أخطاء الإرسال وغالبا ما يستخدم تشفير البيانات لمحاربة أي نظام للتجسس ونحن نحاول أن نقدم نظم الترميز بالمفتاح العمومي باستخدام رموز تصحيح الخطأ وأنماط الهويات

الكلمات المفتاحية

كود هامنغ- امسيلس-نيدراتير-ستارن-فورون-كايرل-كود تصحيح الاخطاء-مخطط التعريف

Abstract

The fast development of global networks and the immense possibilities offered by electronic transactions in continuous communication today pose the problem of information protection against transmission errors on the one hand, and on the other hand must these data is unintelligible except to the audience wanted.

Coding of information to combat transmission errors and data encryption is often used to fight against any spy system.

We try to introduce public key cryptosystems using error correcting codes and patterns of identifications.

Key words

code of hamming- McEliece, Niederreiter, Stern, Veron, Cayrel, code corrector errors- Identification Scheme.

Resume

L'évolution rapide des réseaux mondiaux et les immenses possibilités produites par les transactions électroniques en communication continues, posent aujourd'hui le problème de la protection de données contre les erreurs de transmission d'une part, et d'autre part il faut que ces données soit non compréhensibles sauf à les gens demandé.

Le codage de l'information pour résister les erreurs de transmissions et le chiffrement des données est souvent utilisé pour résister contre tout système d'espionnage.

On essaye d'introduire des cryptosystèmes à clef publique utilisant des codes correcteurs d'erreurs et des schémas d'identifications.

Mots clés

Code de Hamming- McEliece- Niederreiter- Stern- Veron- Cayrel- codes correcteurs d'erreurs- schéma d'identification.