MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE FERHAT ABBAS – SETIF
UFAS(ALGERIE)

MEMOIRE

Présenté à la Faculté des Sciences de l'Ingénieur
Département d'informatique
Pour l'Obtention du Diplôme de

MAGISTER
ECOLE DOCTORALE D'INFORMATIQUE (STIC)

Option : Ingénierie des Systèmes Informatiques (ISI)

Par :

Melle. Kharchi Samia

THEME :

Techniques de Signatures dans une Infrastructure d'Anonymat

Soutenu le : 02 décembre 2009 devant le Jury composé de :

Dr. Allaoua REFOUFI	MCA à l'université de Sétif	Président
Dr. Makhlouf ALIOUAT	MCA à l'université de Sétif	Examinateur
Dr. Mohammed SAIDI	MCB à l'université de Sétif	Membre invité
Dr. Abdellah BOUKERRAM	MCA à l'université de Sétif	Rapporteur

Remerciements

En premier lieu, je tiens à exprimer ma reconnaissance à docteur Abdellah Boukerram, qui m'a été de grandes aides dans la réalisation de ce travail, un grand merci pour sa patience et pour le temps qu'il m'a accordé durant ces deux années. Je remercie aussi docteur Makhlouf Aliouat d'avoir accepté d'être examinateur de ce modeste travail, docteur Allaoua Refoufi d'être le président de mon jury ainsi que docteur Mohammed Saidi qui m'a honoré par sa participation en tant que membre du jury.

Je remercie fortement la source de ma vie de ma joie et de mon bonheur, les chandelles qui m'ont éclairé le chemin du succès : mes parents. Ces deux personnes qui ont tant contribué à ma réussite d'aujourd'hui et de tous les jours.

Un grand merci également à ma nièce, celle que le sourire emplie mes jour d'amour : Maria, à ma sœur Assia, mes deux frère Chérif et Ramzi , sans oublier mon beau-frère Zohir.

Je remercie beaucoup tous les enseignants qui ont participé de loin ou de prés dans notre formations de post-graduation sans exception, particulièrement : Docteur Abdelouaheb Moussaoui, Docteur Abdelhafid Benaouda, Professeur BenMhammed, Docteur Laadjel Bellatreche, Professeur Mohammed Mostefai, docteur Mohammed Touahria et docteur Abdellah Khababa.

Je n'oublie pas les personnes avec qui j'ai passé les plus beaux moments pendant mes études de post-graduation : Lyazid Toumi, Abdelouaheb Attia, abdelaziz Kara, Raouf Lakehal-Ayat, Lotfi Allou, Fateh Seghir, Nouredine Mekroude et Billel Benmessahel.

Enfin mes chères amies : Mouna Hedjeres, Laalama Zahia et Wided Kara qui m'ont été de grand soutient pendant tout ce chemin.

Merci aussi à tous ceux que j'ai oublié et que je sais que la liste est longue.

Table des matières

NOTATIONS

$[a,b]$	Intervalle des entiers entre a et b.
$\{0,1\}^k$	Ensemble des mots de longueur k constitués des caractères 0 et 1.
$a \parallel b$	Concaténation des suites de bites de a et de b.
$a \in_R E$	L'élément a est choisi aléatoirement dans l'ensemble E.
AC	Autorité de Certification.
ACJT	Ateniese, Camenisch, Stadler, Tsudik : un schéma de signature de groupe proposé en 2000.
AES	Advenced Encryption System : Cryptosystème symétrique proposé en 2000.
BSZ	Bellare, Shi, Zhang : un schéma de signature de groupe proposé en 2005.
CBC	Cipher Bloc Chaining : Mode opératoire d'un chiffrement symétrique.
DES	Data Encryption Standard : Cryptosystème symétrique proposé en 1977.
DSA	Digital Signature Algorithm :Algorithme de signature numérique
IdE	Identity Escrow : Séquestre d'identité.
IGC	Infrastructure de Gestion de Clés.
a^{-1}	L'inverse multiplicatif de a modulo un nombre entier.
$\mathbb{N}$	Ensemble des entiers naturels.
PK	Public Key : Clé publique.
PKC	Public Key Cryptosystem : Cryptosystème à clé publique.
PKI	Public Key Infrastructure.
RSA	Rivest, Shamir, Adleman : Cryptosystème asymétrique proposé en 1978.
SK	Secret Key : Clé secrète.
$\mathbb{Z}$	Ensemble des entiers relatifs
$\varphi(n)$	L'indicateur d'Euler : $\varphi(n)= (p-1)(q-1)$ avec $n = p * q$

INTRODUCTION

La cryptologie est une science très ancienne, ayant pour essence l'art de cacher une information au sein d'un message chiffré. Mais elle a véritablement éclos avec l'avènement de l'informatique et la généralisation des télécommunications. Longtemps fermée, réservée aux militaires et à la diplomatie, elle touche tout le monde depuis l'explosion d'Internet.

La cryptologie a connu une mutation importante avec l'apparition de la cryptographie dite à clé publique, qui a ouvert un domaine de recherche très riche avec tous les problèmes pratiques de sécurité : authentification, confidentialité, commerce, vote, enchères et toutes autres formes d'échange de données nécessitant l'intégrité, la non-répudiation ou l'anonymat.

Ce dernier concept qui peut être défini comme la propriété qu'un utilisateur ne puisse pas être identifié parmi un ensemble de sujets est appelé *groupe anonyme*. Il est important de remarquer que cette définition de l'anonymat n'est pas absolue mais relative à un groupe, en particulier à sa taille. Ainsi, il est très différent d'être anonyme au sein d'un groupe de deux, de cent ou de dix-mille utilisateurs.

A partir de cela, on peut définir plusieurs types d'anonymat pour les communications. L'*anonymat d'émissio*n est obtenu par un utilisateur face à un attaquant lorsque celui-ci est incapable de détecter l'émission de messages par l'utilisateur. De manière similaire, l'*anonymat de réception* est obtenu par un utilisateur face à un attaquant lorsque celui-ci est incapable de détecter la réception de messages par l'utilisateur. L'*anonymat relationnel* est obtenu par un groupe d'utilisateurs face à un attaquant lorsque l'attaquant est incapable de savoir si deux membres du groupe communiquent entre eux ou pas. De ces définitions, il découle de façon immédiate que si l'anonymat d'émission (ou de réception) est garanti à chaque utilisateur d'un groupe, le groupe possède la propriété d'anonymat relationnel.

Il est important de remarquer que les anonymats d'émission, de réception et relationnel ne procurent pas par eux-mêmes la confidentialité du contenu des messages : il faut pour cela utiliser des techniques appropriées, comme le chiffrement. À l'inverse, le chiffrement du contenu des messages en soi ne procure pas l'anonymat des communications.

Mais l'anonymat total peut mettre en péril l'économie mondiale ou la sécurité des individus, et donc, dans certains domaines (monnaie électronique, télécommunications, ...) Une question se pose : est-il souhaitable d'instaurer un anonymat contrôlable ? Ce qui est appelé communément : *anonymat révocable*.

Les questions d'anonymat surgissent en de nombreuses occasions en télécommunications. Tout d'abord dans le contexte du commerce électronique, et plus particulièrement pour le porte-monnaie électronique qui est censé remplacer l'argent liquide. Une autre application de l'anonymat concerne les communications entre interlocuteurs mobiles (ex : la téléphonie mobile, mais aussi les communications grâce à Internet). Enfin, une autre application, qui a de tout temps été synonyme d'anonymat, n'est autre que le vote ou le suffrage universel

Comme il a été signalé précédemment, autour du chiffrement, on trouve la signature parmi les techniques d'anonymisation. Cette technique a pour principe d'être

enregistré auprès d'une autorité de certification qui peut fournir une paire unique de clés (clé privée et clé publique) ainsi que la possession d'un protocole servant au chiffrement (RSA, PGP). Ce qui garantie une unicité de signature et une préservation de l'identité du signataire.

Dans ce contexte ; on peut définir trois (03) types de signature pour l'anonymat : les *signatures de groupe* ; qui est un procédé cryptographique permettant a un utilisateur de signer anonymement au nom d'un groupe. Ainsi le destinataire de la signature a l'assurance que celle-ci a été produite par l'un des membres du groupe mais ne sait rien de plus. Généralement, un groupe est composé de membres et d'un manager de groupe chargé de la gestion des membres. En cas de contestation, l'anonymat de la signature peut être levé par une autorité de confiance, en utilisant des trappes spécialement introduites dans les protocoles (la signature peut être ouverte, avec ou sans l'aide des membres du groupe, et révéler l'identité du signataire.).

La *signature aveugle*, étant le deuxième type. Cette signature n'a pas pour objectif de masquer l'identité du signataire, mais de masquer le contenu du message aux yeux de ce dernier. Plus précisément, l'utilisateur obtient une signature sur un message de son choix, en interagissant avec un signataire, sans que celui-ci ait connaissance du message. Cette dernière trouve de nombreuses applications en cryptographie, comme le vote ou la monnaie électronique.

Enfin, les *signatures d'anneau*, comme les signatures de groupe, permettent à un utilisateur de signer anonymement, au nom d'un ensemble de personnes, de telle sorte que le destinataire ne puisse identifier le membre à l'origine de la signature (il sait de quel groupe/ anneau provient la signature, mais ne reconnaît pas le signataire), sauf que le signataire n'est pas obligatoirement membre d'un groupe de signature. Lorsqu'un utilisateur veut signer un message, il construit lui même son anneau, en choisissant librement les membres qui l'intéressent. Il peut même changer d'anneau pour toute nouvelle signature. La seule condition requise pour être membre d'un anneau est d'avoir publié une clé publique de signature (une clé RSA par exemple). Contrairement aux signatures de groupe, il n'existe pas de procédure de révocation pour les signatures d'anneau. Elles fournissent donc un anonymat inconditionnel pour le signataire.

Ce mémoire est agencé autour de deux parties, dictées par la progression dans le domaine cryptographique: les préliminaires cryptographiques, l'anonymat et les signatures anonymes.

Dans la première partie, consacré aux concepts de base de la cryptographie, le chapitre 1 présente les notions générales cryptographiques, suivies des objectifs et des différentes attaques contre un cryptosystème. Le chapitre suivant explique bien les cryptosystèmes symétriques et les cryptosystèmes asymétriques. Dans le troisième chapitre, les signatures électroniques sont présentées, leurs: principes, les attaques possibles. Des exemples de schémas de signature électronique illustrent bien ces questions.

La deuxième partie est consacrée au domaine d'anonymat, où sont définies, les notions des techniques d'anonymat suivies des techniques de signatures anonymes existantes. Le chapitre 4 présente, la signature de groupe, tout en mettant en relief les célèbres schémas de signature d'aujourd'hui puis les signatures aveugles et leur application au vote et à la monnaie électronique.

Et c’est au chapitre 5 que nous avons étudié en générale les signatures d’anneau, leur principe et leur fonctionnement. Après nous avons formulé une description plus au moins détaillée du schéma de Rivest et al. [RST01] qui présente la base du modèle que nous avons proposé.

Chapitre 1 :

Préliminaires Cryptographiques

1.1. Définitions

La cryptologie est un concept grec composé de « kryptos » et « logos » qui signifie « la connaissance de ce qui est caché », qui est devenu une discipline; dont les origines remonteraient au XVIème siècle avant J-C. Elle peut être définie comme étant la science des messages secrets. Restreinte depuis longtemps aux usages diplomatiques et militaires, elle est maintenant une discipline scientifique à part entière, dont les applications sont si vastes aujourd'hui qu'il est difficile de définir a priori ce qui révèle ou non de la cryptologie [Mol07].

Cette science se divise en deux branches étroitement liées, opposées mais complémentaires : celle qui consiste à protéger l'information « la cryptographie », et l'autre qui cherche à étudier la sécurité apportée par les mécanismes de la cryptographie et la faire échouer « la cryptanalyse ».

1.1.1. Cryptographie

La cryptographie est l'étude des méthodes mathématiques liées aux aspects de sécurité de l'information, tel que : la confidentialité, l'intégrité et l'authentification, de manière à transformer des informations compréhensibles (textes claires) en informations inintelligibles (textes chiffrés) pour des tierces n'ayant pas la connaissance du secret, d'un coté, ou d'inverser l'opération d'un autre coté à l'aide d'outils spécialement conçus à cet effet (fonctions et clés).

D'un autre angle, la cryptographie peut être vue comme l'étude d'un protocole sécurisé d'échange d'informations, utilisant certaines données qui doivent rester secrètes entre les parties communicantes, dont on peut citer : les clés de cryptage et la fonction utilisée. Ce protocole est communément connu sous le nom de cryptosystème.

1.1.2. Cryptosystème

Un cryptosystème est défini par un quintuplet :

$$P=(M,C,K,E,D).$$

Où :

M : l'ensemble des textes clairs ;

C : l'ensemble des textes chiffrés ;

K : l'ensemble des clés ;

E : l'ensemble des fonctions de chiffrement (les fonctions chargées de transformer les textes clairs en textes chiffrés) ;

D : l'ensemble des fonctions de déchiffrement (l'opération inverse du chiffrement).

1.2. Concepts de base de la cryptologie

1.2.1. Chiffrement :

Le chiffrement est la transformation de l'information échangée de telle sorte que seule le destinataire légitime pourra la visualiser. Il se divise en deux axes différents, selon la technologie choisie : chiffrement symétrique et chiffrement asymétrique.

✓ *Chiffrement symétrique :*

Dit aussi : cryptosystème à clé secrète. C'est la plus ancienne des techniques, qui consiste à utiliser la même clé (dite clé de session) pour le chiffrement ainsi que pour le déchiffrement.

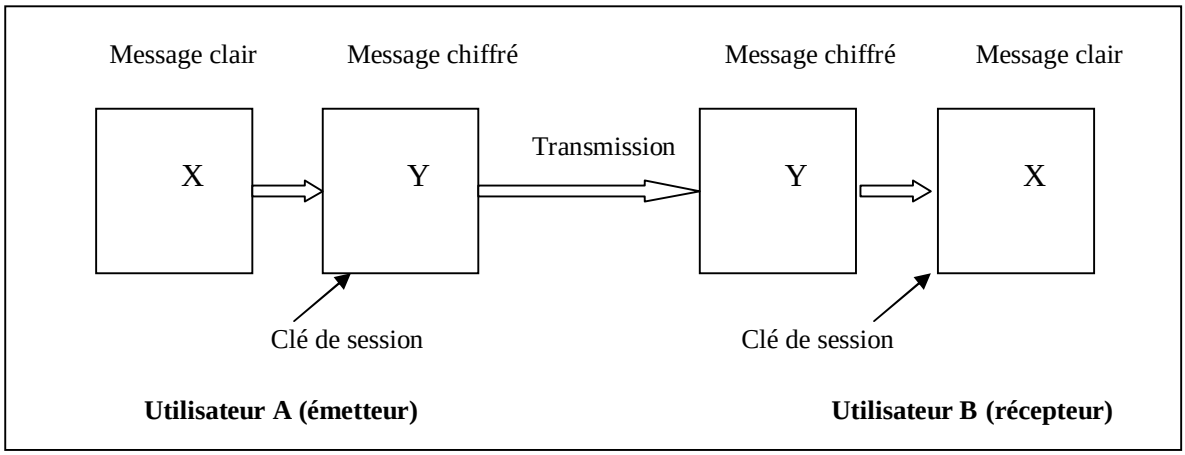


Figure 1 : ***Chiffrement symétrique***

Ce chiffrement a pour inconvénients :

- Le devoir de partager la même clé entre utilisateurs alors que sa transmission elle-même n'est pas sécurisée.

- La nécessité d'avoir plus d'une clé à chaque utilisateur pour pouvoir communiquer avec différents groupes en privé.

✓ *Chiffrement asymétrique :*

Il est dit : chiffrement à clé publique : la clé de chiffrement (clé publique) diffère de celle de déchiffrement (clé privée), et aucune d'entre elles ne pourra être découverte à partir de l'autre. Cette paire de clés est dite indissociable : le texte chiffré avec l'une ne pourra pas être déchiffré que par l'autre.

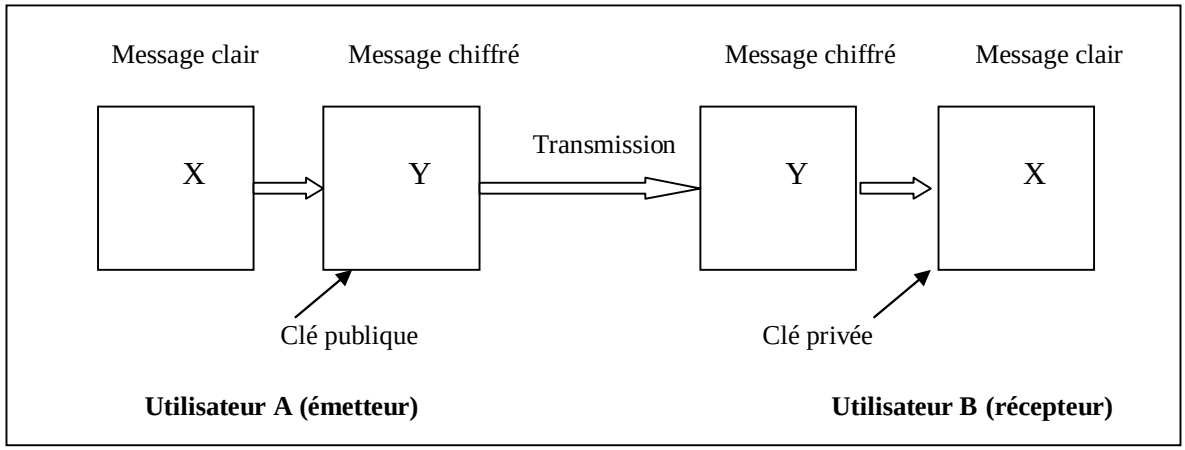


Figure 2 : ***Chiffrement asymétrique***

Ce type de chiffrement résout le problème de sécurité d'échange de clé car la clé publique n'est pas nécessairement confidentielle (généralement, publiée sur une base de données réservée à ce genre d'opérations).

1.2.2. Hachage :

Le hachage est une fonction mathématique qui consiste à compresser un message de longueur variable et donner une empreinte appelée condensé (condensat haché), qui est d'une taille beaucoup plus petite que le message lui-même.

Les fonctions de hachage doivent satisfaire certaines conditions [Mov96]:

- *Fonction à sens unique* : cela signifie que : étant donné le haché, il est calculatoirement difficile d'établir le message initial.
- *Fonction résistante aux collisions* : il est difficile de trouver une même empreinte pour deux messages différents (le haché représente de façon unique le document à partir duquel il a été calculé).
- *Fonction résistante à la seconde préimage* : il n'existe pas d'algorithme efficace qui pourra trouver deux messages différents ayant la même empreinte (La fonction de hachage est telle qu'il est pratiquement impossible statistiquement de produire deux messages dont les condensés seraient identiques).

La technique de hachage est utilisée pour préserver l'intégrité d'une information en envoyant cette information accompagnée de son haché. Ce dernier étant éventuellement chiffré. Le message et son condensé sont donc liés : on ne peut pas modifier l'un sans devoir modifier l'autre. Les fonctions de hachage les plus connues sont MD2, MD5 et SHA.

1.2.3. Signature numérique :

C'est un procédé cryptographique qui assure l'authentification des origines des informations échangées ainsi que sa non-répudiation, il garantie aussi l'intégrité des informations grâce à la combinaison des deux techniques précédemment citées (chiffrement et hachage).

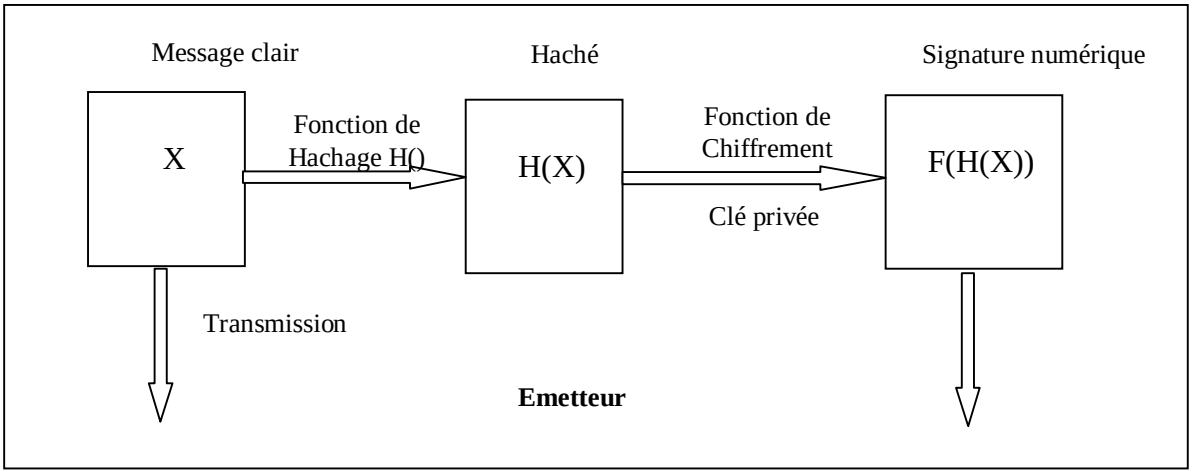


Figure 3 : ***signature numérique (coté émetteur)***

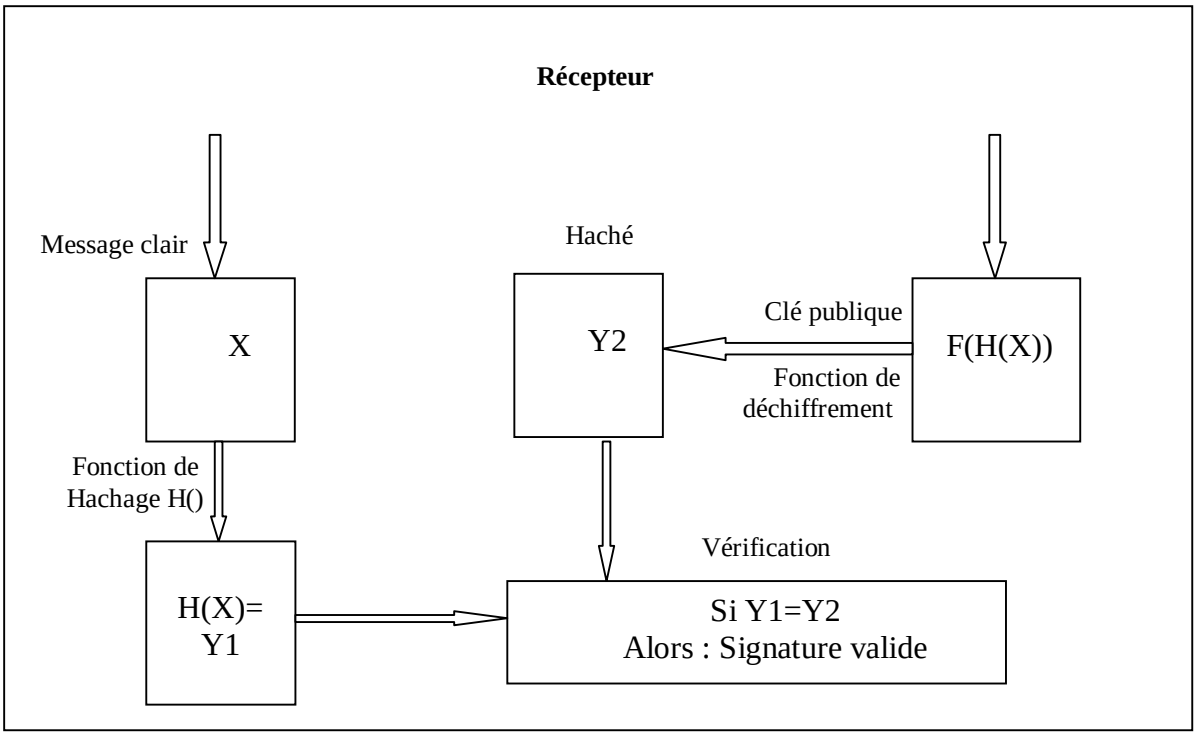


Figure 4 : ***Vérification de la signature (coté récepteur)***

Les figures ci-dessus (figure 3 et figure 4) illustrent une technique de signature électronique. Cette signature consiste à hacher le message puis le chiffrer à l'aide de la clé privée de l'émetteur, pour l'envoyer (le haché signé) accompagné du texte original à son destinataire qui aura besoin de la clé publique de l'émetteur afin de déchiffrer le haché signé et le comparer au message original après l'avoir haché à son tour (avec bien entendu, le même algorithme de hachage que celui, utilisé la première fois).

S'il y a nécessité de confidentialité, ce qui n'est pas garantie ici, l'émetteur doit aussi chiffrer le message original avec la clé publique du destinataire afin de s'assurer que seul le récepteur légitime pourra le déchiffrer.

1.3. Les objectifs de la cryptographie

Une méthode cryptographique sûre doit répondre à certaines propriétés, dites les propriétés de sécurité, prédéfinies comme objectifs [Mov96]:

1.3.1. La confidentialité

Historiquement, la confidentialité est l'un des plus anciens objectifs de la cryptographie. C'est un service utilisé pour cacher le contenu des messages de toute personne non autorisée, à l'exception de son réel destinataire.

Donc, un cryptosystème est choisi afin de transformer le message de sa forme intelligible ; utilisant une clé et un algorithme de chiffrement ; en une forme inintelligible pour une tierce personne qui tente de l'intercepter et qui doit avoir connaissance de l'algorithme utilisé et la clé de déchiffrement pour pouvoir le lire.

1.3.2. L'intégrité

Si jamais le message envoyé chiffré tombe entre les mains d'une tierce personne malveillante, qui a connaissance de la clé de déchiffrement, il découvre son contenu, le modifie, le chiffre de nouveau puis le renvoie à son destinataire. Ce dernier ne pourra pas découvrir que le message a été altéré en route. Et la cryptographie rencontre un nouveau défit à assurer à savoir, l'intégrité.

A fin de défier les malfaiteurs, les cryptographes utilisent une nouvelle technique, dite « le hachage » (voir la section suivante) pour garantir l'intégrité des échanges.

1.3.3. L'authenticité

Cette notion regroupe deux choses importantes : l'authentification des données qui permet de garantir que les données transmises proviennent bien d'un émetteur désigné, et l'authentification de l'émetteur lui-même qui consiste à prouver son identité.

L'authenticité est, alors, faite de façon interactive ou non interactive.

Le scénario non interactif pour l'authentification des données utilise la signature numérique simple qui nécessite la connaissance d'une clé privée pour signer le document et elle est vérifiable par tout le monde en possession de la clé publique.

L'authentification de l'émetteur, repose sur le scénario interactif, qui est un dialogue entre deux entités, où le prouveur montre sa connaissance d'un secret, similaire à un contrôle

d’accès par mot de passe, comme par exemple montrer au vérificateur la connaissance de la clé privée couplée à une clé publique sans avoir recours à dévoiler sa clé privée (le concept de preuve de connaissance).

1.3.4. La non-répudiation

La non-répudiation permet d'obtenir la preuve de l'émission d'une information ou la preuve de sa réception. L'émetteur ou le récepteur ne peut ainsi en nier l'envoi ou la réception. Ce service peut être garanti avec le concept de la signature électronique. Où, seul le vrai émetteur peut signer avec sa clé privée et seul le vrai destinataire peut également le déchiffrer avec sa clé privée

1.3.5. L’anonymat

La problématique de l’anonymat concerne aussi bien les données directement nominatives (nom, prénom, date de naissance,…) que celle qui le sont indirectement (matricule, adresse, N° de téléphone,…). Alors l’anonymat consiste en la possibilité de suivre une personne unique dans la durée (caractéristiques, comportements, …) sans avoir la moindre possibilité de connaître sa véritable identité. Sur le plan technique, la démarche consiste à transformer des données nominatives via un algorithme de hachage permettant d’obtenir un identifiant (a priori) unique et surtout irréversible.

1.4. Les attaques contre les cryptosystèmes

Une attaque contre un cryptosystème est tout processus de recherche d’informations de transformation du texte chiffré en un texte claire. Le but final d’une telle méthode est de déterminer la clé privée de déchiffrement ainsi que le texte en claire.

On peut déterminer deux types d’attaques : les attaques passives où le cryptanalyste détourne le canal de communication (attaque contre la confidentialité de données), et attaques actives : où le cryptanalyste cherche à modifier le message (ajout ou suppression d’informations) ; autrement dit : attaque contre l’intégrité et l’authenticité des données [Mol07].

1.4.1. Les attaques passives

Les attaques passives peuvent être subdivisées en quatre classes principales:

1.4.1.1. Les attaques à texte clair choisi

Avec cette attaque, le cryptanalyste choisi un texte clair, il le chiffre avec une clé publique de chiffrement donnée puis, il analyse les résultats afin de déduire certaines informations pouvant lui servir à découvrir le texte clair correspondant à un chiffré non vu précédemment.

1.4.1.2. Les attaques à texte chiffré choisi

Dans cette attaque, le cryptanalyste choisi un texte chiffré, et trouve son texte en clair, correspondant. Cela ne peut se faire sans avoir accès aux équipements de chiffrement. Après, le cryptanalyste peut utiliser de tel équipement pour déduire le texte clair correspondant à un chiffré intercepté, puisque l'équipement ne donne pas la clé de déchiffrement.

1.4.1.3. Les attaques à texte clair connu

Cette attaque est plus pratique que celle à texte clair choisi, car l'attaquant a un certain nombre de paires constituées de texte clair et de texte chiffré. Même une petite quantité d'information dans ce type d'attaque suffira pour trouver la clé.

1.4.1.4. Les attaques à texte chiffré seul

L'attaquant ne connaît que le texte chiffré et il cherche le texte clair et si possible la clé. Tout cryptosystème vulnérable à ce type d'attaque est considéré non sûr.

1.4.2. Les attaques par force brute (exhaustives)

L'attaquant essaie toutes les clés possibles afin de déterminer laquelle a été utilisée par les parties communicantes, dans le but d'avoir la clé de déchiffrement pour des fins de modification. Ce type d'attaque peut prendre un temps considérablement long.

Chapitre 2 :

LES CRYPTOSYSTEMES SYMETRIQUES ET ASYMETRIQUES

2.1. Les cryptosystèmes symétriques

Le principe des cryptosystèmes symétriques est de pouvoir envoyer des messages d'une longueur éventuellement grande de plusieurs milliers de caractères en utilisant une clé secrète à longueur fixe, ne dépassant pas quelques centaines de bits.

Dans cette partie, nous allons étudier, certains protocoles qui historiquement parlant, sont les plus célèbres :

2.1.1. Les cryptosystèmes alphabétiques

Ils sont appelés également, chiffrement par substitution. Ce type de chiffrement consiste à remplacer des caractères par d'autres caractères afin de construire le texte chiffré [Mol07].

Cette notion de chiffrement remonte à des milliers d'années, où Julius César a chiffré ces documents en remplaçant chaque caractère par le troisième caractère à sa droite dans l'alphabet. Dans le schéma de César, le chiffrement se fait par le décalage de chaque caractère à travers un certain nombre de position dans l'alphabet. Et le déchiffrement décale chaque caractère à travers le même nombre de positions dans le sens inverse.

A titre d'exemple, le chiffrement du mot CRYPTOGRAPHIE donne le cryptogramme: FUBSWRJUDSKLH

La clé secrète dans ce schéma est le nombre de positions à travers lequel les caractères ont été décalés. Autrement, la clé est la table de correspondance suivante (figure 5) :

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Figure 5 : ***table de chiffrement et déchiffrement de César***

Où chaque lettre est substituée par une et une seule lettre, alors le chiffrement de César est dit chiffrement par substitution mono-alphabétique.

Sans cette table, le chiffrement semble être impossible, mais les cryptanalystes essaient toutes les clés possibles jusqu'à avoir un texte significatif en utilisant les 26 substitutions possibles. Alors, on dit que le cryptosystème de César peut être facilement cassé par les attaques à force brute.

A partir de ces insuffisances, d'autres cryptosystèmes ont été développés.

A. Le chiffrement de Vigenère :

Pour un cryptosystème plus sécurisé, Blaise de Vigenère proposa son schéma. Il créa une nouvelle notion pour la clé : un mot-clé.

Au lieu de faire de simples rotations de l'alphabet, le chiffrement de Vigenère associe à chaque lettre un numéro représentant son ordre dans l'alphabet (A=1, B=2,…, Z=26) puis additionne la valeur de chaque lettre du texte claire à sa valeur correspondante du mot-clé pour obtenir le texte chiffré.

Si la valeur résultat est plus grande que 26, alors on soustrait 26 de cette valeur et la valeur résultat donne le caractère du cryptogramme.

Cette opération est répétée autant de fois nécessaire que la longueur du texte claire exige.

Exemple : soit le même mot : CRYPTOGRAPHIE avec la clé : SECRET.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

Figure 6 : ***table de numérotation des lettres***.

Le texte claire :	C	R	Y	P	T	O	G	R	A	P	H	I	E
Correspondance :	03	18	24	16	20	15	07	18	01	16	08	09	05
La clé :	S	E	C	R	E	T	S	E	C	R	E	T	S
Correspondance :	19	05	03	18	05	20	19	05	03	18	05	20	19
Résultat :	22	23	01	08	25	09	26	23	04	08	13	03	24
Le cryptogramme :	V	W	A	H	Y	I	Z	W	D	H	M	C	X

Figure 7 ***: chiffrement du mot CRYPTOGRAPHIE par Vigenère***

Dans ce cas le cryptogramme est VWAHYIZWDHMCX.

Cette solution peut substituer une lettre en plusieurs lettres selon son emplacement par rapport à la clé. Dans ce cas, ce chiffrement est appelé : chiffrement par substitution poly-alphabétique.

Mais cette solution ne résiste pas non plus aux tentatives des cryptanalystes, qui réussissaient à la mettre en défaite.

2.1.2. Les chiffrements par blocs : le DES

Ce nouveau procédé de chiffrement manipule cette fois des mots binaires (afin de pouvoir exécuter des opérations du ou-exclusif (XOR) entre le texte en clair et la clé ; le résultat sera toujours un caractère bien défini dans l'alphabet utilisé. Il consiste à découper le texte en clair en blocs de taille constante et chaque bloc est chiffré indépendamment des autres blocs [Fun05] [MOV96].

Figure 8 : ***chiffrement par blocs***

Dans le cas du DES (Data Encryption System), il manipule des blocs de 64bits en utilisant une clé de 56bits (pour compléter la clé à 64bits, une information supplémentaire est ajoutée et qui peut être connue sans difficulté comme par exemple : 8bits de parité).

En utilisant la même clé sur les différents blocs et sachant qu'un bloc peut se répéter plusieurs fois, cela peut donner des blocs chiffrés similaires qui pourra aider les cryptanalystes à casser ce cryptosystèmes. Alors un nouveau mode d'utilisation a été introduit : le CBC.

- ✓ **CBC (Cipher block Chaining)** où le nouveau cryptogramme d'un bloc ne dépend pas seulement du texte en clair et de la clé mais aussi du cryptogramme du bloc déjà chiffré précédemment (le bloc qui précède le bloc actuel, et pour le premier bloc, un vecteur d'initialisation est utilisé). De cette manière, le texte en clair ne donne jamais deux fois le même cryptogramme.

- ✓ **Le triple DES (3-D.E.S)**

La longueur de la clé du DES était un vrai défi pour les cryptanalystes. Mais vu le développement des pouvoirs calculatoires des ordinateurs, une clé de 56 bits n'est plus d'une grande force à battre. A cela, le triple DES fait apparition. Ayant les mêmes caractéristiques du DES, sauf qu'il fait trois opérations : un chiffrement, un déchiffrement et un chiffrement. C'est avec une clé de 168bits (3*56) qu'il procède.

Pour générer un cryptogramme, l'émetteur chiffre le texte en clair avec la première clé, il le déchiffre ensuite avec la deuxième clé puis le chiffre avec la troisième clé.

Le destinataire, alors, déchiffre le cryptogramme reçu à l'aide de la troisième clé, chiffre le résultat avec la deuxième clé pour finalement le déchiffrer avec la première clé.

2.1.3. Le standard de chiffrement A.E.S :

Le DES étant obsolète et le 3-DES trop couteux en ressources, le NIST a mis au point une nouvelle norme : le standard de chiffrement avancé AES (Advanced Encryption Standard) qui opère avec trois tailles possibles de clé : 128, 192, 256 en remplacement du

DES. Il est capable d'opérer sur différentes plateformes et facilement implémentable sur des cartes à puce [Fun05].

2.2. Cryptosystèmes asymétriques

Les années 70 ont vu un changement révolutionnaire dans la manière de manipulation des clés cryptographiques. C'était dans le sens de rendre la cryptographie publique.

L'idée est parvenue de Whitfield Diffie et Martin Hellman [DH76], qui ont conçu une méthode, pour deux entités qui ne se sont jamais rencontrés avant ou échangé de clés, d'établir une clé secrète partagée en échangeant des messages sur un canal ouvert (non sécurisé).

Avant cette idée, tous les cryptosystèmes, y inclus DES, cherchaient un mécanisme sécurisé de distribution de clés secrètes. Car c'est connu que si la clé de chiffrement symétrique est découverte, la déduction de la clé de déchiffrement est facile, et le cryptosystème n'est plus sécurisé. Maintenant, avec l'introduction de l'idée de Diffie-Hellman, connue par : l'échange de clé Diffie-Hellman, deux entités peuvent échanger des clés d'une manière confidentielle sur un canal non sécurisé.

2.2.1. Cryptosystème à clé publique (PKC)

Un cryptosystème à clé publique est défini par trois algorithmes :

- PKC.KGen : c'est un algorithme de génération de clés qui prend en entrée un paramètre de sécurité « L » (longueur de la clé) et donne une paire de clés « sk, pk ».

- PKC.Enc : c'est un algorithme de chiffrement qui prend en entrée un message M à chiffrer et la clé publique pk du destinataire, et donne en sortie un cryptogramme C.

- PKC.Dec : un algorithme de déchiffrement qui prend en entrée un message chiffré C et la clé privée du destinataire et donne en sortie le message en clair M.

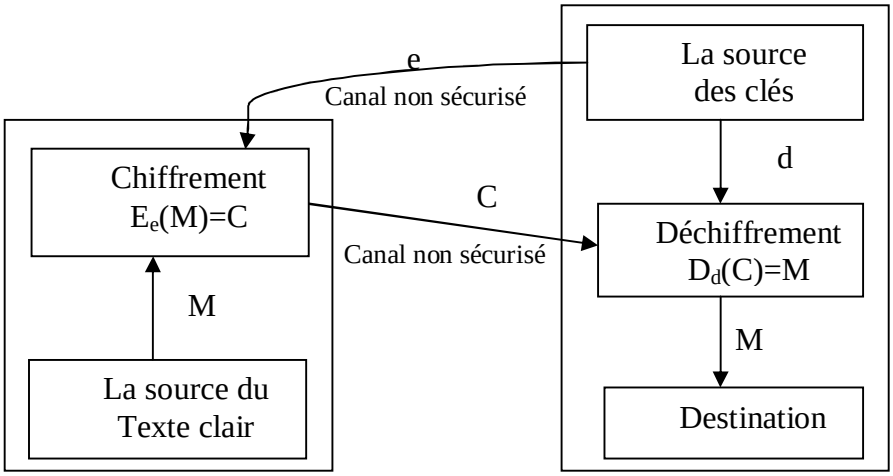


Figure 9 : ***Un cryptosystème asymétrique***

2.2.2. L'échange de clés de Diffie-Hellman

Algorithme créé en 1976 par Diffie et Hellman, il est utilisé pour l'échange de clé à travers une liaison non sécurisée, et non pour le chiffrement et le déchiffrement des messages. Il repose sur la difficulté du problème de logarithme discret. Néanmoins, sa sécurité dépend aussi de la taille de l'exposant secret [DH76].

Son principe est d'établir un accord sur certains paramètres :

- p : un nombre entier premier.
- g : un générateur d'un groupe multiplicatif qui doit être inférieur à p (g est dit générateur si et seulement si il peut générer chaque élément de l'intervalle [1..p-1] quand il est multiplié par soit même un certain nombre de fois, le résultat modulo p).

Ces deux paramètres sont rendus publics.

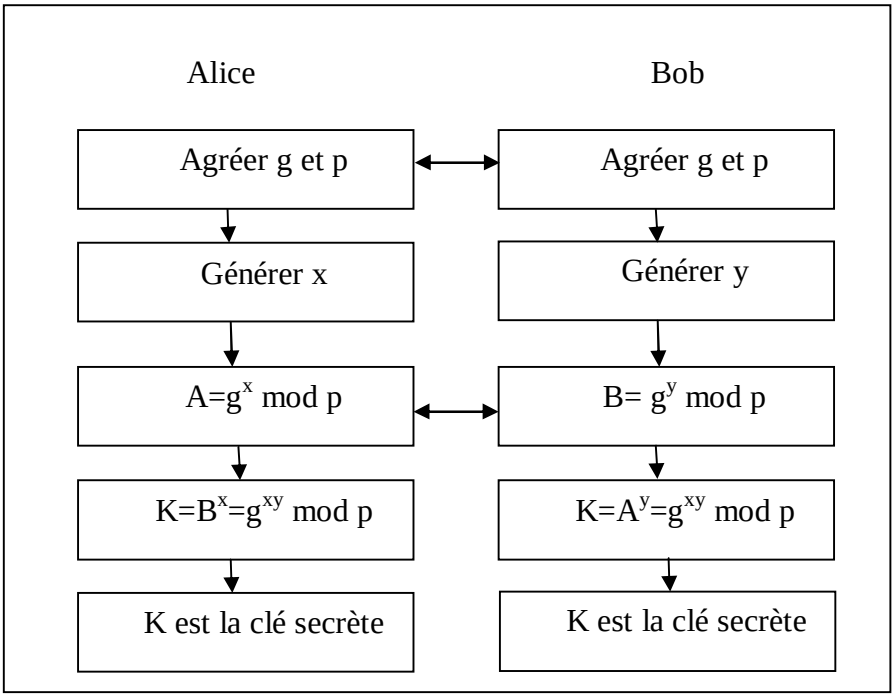


Figure 10 : ***Echange de clé Diffie-Hellman***.

La figure 10 illustre l'algorithme, où Alice et Bob sont en accord sur p et g. Alice génère un nombre aléatoire x, calcule :

$$A=g^x \bmod p$$

et l'envoie à Bob, qui, lui aussi génère un nombre aléatoire y, calcule :

$$B= g^y \bmod p$$

et l'envoie aussi à Alice.

Maintenant Alice calcule :

$$K=g^{xy} \bmod p$$

et Bob fait le même calcule. K est la clé secrète partagée qui servira ensuite au chiffrement et déchiffrement entre Alice et Bob. Et comme x et y sont propriétaires à Alice et Bob et personne d'autre ne les connaît, un attaquant ayant g et p (publics) et A et B (interceptés sur le canal de transmission) ne pourra jamais calculer la clé K.

Donc une clé secrète a été échangée avec succès sans avoir à l'envoyer sur le réseau.

2.2.3. Le cryptosystème d'El-Gamal

Portant le nom de son concepteur, le cryptosystème El-Gamal permet l'échange sécurisé des messages chiffrés. Comme tout cryptosystème à clé publique, il comporte les trois algorithmes suivants :

- **La génération de clé El-Gamal** : l'algorithme de génération des clés est le suivant :
  - Bob choisi un nombre premier p (aussi large qu'il peut) et générateur g modulo p.
  - Il choisi ensuite aléatoirement un entier a ϵ [2..P-1] et calcule :

    $$g^a \bmod p.$$

  - La clé publique de Bob est (p, g, g^a) et sa clé privée est a.
- **Le chiffrement El-Gamal :** Le chiffrement comporte trois étapes :
  - Alice obtient la clé publique de Bob (p, g, g^a).
  - Elle choisi un nombre b<p-1 aléatoirement.
  - Elle calcule ensuite :

    $$g^b \bmod p \quad \text{et} \quad mg^{ab} \bmod p.$$

  Elle envoie alors le cryptogramme

  $$c= (g^b, mg^{ab})$$

  à Bob.
- **Le Déchiffrement El-Gamal :**

  Bob ayant reçu le cryptogramme c , il procède comme suit :

  - Il utilise sa clé privée pour calculer :

$$(g^b)^{-a} \equiv (g^b)^{p-1-a} \bmod p.$$

- Il déchiffre ensuite m en calculant :

$$(g^b)^{-a} m g^{ab} \bmod p.$$

2.2.4. Le cryptosystème RSA

RSA est probablement le premier cryptosystème à clé publique, le plus fréquemment utilisé**,** proposé par Rivest, Shamir et Adleman en 1978. Il repose sur la quasi-impossibilité d'inverser une fonction puissance et est considéré comme sûr quand la clé est suffisamment longue.

Il permet d'utiliser des clés de :

- 512bits : peu sûr
- 768bits : légèrement sûr
- 1024bits : sûr pour la plupart des utilisations
- 2048bits : sûr encore pour 10 ans.

Sa sécurité, alors, repose sur la taille de sa clé (la difficulté de factorisation des grands nombres entiers).

Le cryptosystème RSA utilise les trois algorithmes suivants :

- **La génération des clés RSA**

Comme déjà mentionné avant, la force de RSA réside dans la taille de ces clés et le choix d'un aussi grand nombre entier n qui va être factorisé en p et q. alors la solution est d'agir dans le sens inverse : on commence par p et q.

- Bob doit choisir d'abord aléatoirement deux nombres premiers suffisamment grands ayant à peu près la même taille et calculer

$$n = p * q \quad \text{et} \quad \varphi(n) = (p-1)(q-1).$$

- Il choisi ensuite aléatoirement aussi un nombre naturel $e \in N$ tel que

$$1 < e < \varphi(n) \quad \text{et} \quad \text{pgcd}(e, \varphi(n)) = 1 \ (\text{e : est premier avec } \varphi(n)).$$

- En utilisant l'algorithme d'Euclid étendu, il calcule

$$d \in N \quad \text{tel que} \quad 1 < d < \varphi(n) \quad \text{et} \quad ed \equiv 1 \bmod \varphi(n).$$

- Il publie alors sa clé publique (n , e)dans une base de données et garde d, p, q et $\varphi(n)$ privés.

n : est appelé le module de RSA

e : l'exposant de chiffrement

d : l'exposant de déchiffrement.

- **Le chiffrement RSA**

Le message m à chiffrer étant en forme numérique, Alice peut commencer :

- Elle cherche la clé publique de Bob (n , e) dans la base de données (annuaire spécialisé).
- Elle chiffre le message m en calculant :

$$c \equiv m^e \bmod n.$$

- Elle envoie le message à Bob.

- **Le déchiffrement RSA**

Une fois le cryptogramme reçu, Bob utilise sa clé privée d et calcule

$$m \equiv c^d \bmod n.$$

2.3. Cryptographie symétrique vs. Cryptographie asymétrique

Cryptographie symétrique ou cryptographie asymétrique, laquelle choisir ? Le choix est difficile surtout entre des cryptosystèmes qui ont prouvé leur force contre les différentes attaques et ceux qui ont des points communs.

Cette partie souligne les principaux avantages et inconvénients des deux techniques.

2.3.1. Les avantages des cryptosystèmes symétriques

- Les chiffrements symétriques peuvent être conçus pour avoir des taux élevés du flux de données. Certaines implémentations matérielles réalisent un taux de chiffrement de centaines de méga-octets par seconde, tandis que les implémentations logicielles peuvent atteindre des taux de sortie en méga-octets par microsecondes.

- Les clés de chiffrement sont relativement courtes (de petite taille).

- Les cryptosystèmes symétriques peuvent être utilisés comme primitives de construction de mécanismes cryptographiques comme la signature électronique.

- Ils peuvent être composés de façon à construire des cryptosystèmes plus forts (utilisation de DES en trois étapes pour le 3-DES).

2.3.2. Les inconvénients des cryptosystèmes symétriques

- La clé doit rester secrète entre deux entités communicantes.

- Sur un grand réseau de communication, il doit y avoir plusieurs paires de clés.

- Dans une communication bipartite entre les entités A et B, une pratique cryptographique saine exige que la clé doit changer fréquemment, et peut-être pour chaque session de communication.

2.3.3. Les avantages des cryptosystèmes asymétriques

- Seule la clé privée doit être gardée secrète.

- La paire de clés (clé publique/clé privée) peut rester inchangeable pour une période considérablement longue.

- Certains schémas de cryptographie asymétrique peuvent produire des mécanismes de signature électronique relativement efficace.

- Sur un réseau suffisamment large, le nombre de clés nécessaires peut être plus petit que le nombre exigé dans les scénarios à clé secrète.

2.3.4. Les inconvénients des cryptosystèmes asymétriques

- La plus populaire des méthodes à clé publique est très lente par rapport aux cryptosystèmes symétriques.

- La taille des clés doit être plus grande que celle des cryptosystèmes symétriques.

Pour conclure cette comparaison, on peut dire que ces deux concepts peuvent être complémentaires, comme par exemple : la cryptographie asymétrique peut servir à l'échange de clé symétrique entre deux entités communicantes A et B. Et pour être juste avec les deux techniques, on peut dire que chacune est meilleure dans un domaine particulier :

- La cryptographie à clé publique est efficace pour les signatures (non-répudiation) et la gestion des clés, et

- La cryptographie à clé symétrique est efficace pour la confidentialité et l'intégrité des données.

Chapitre 3 :

LA SIGNATURE ELECTRONIQUE

Un schéma de signature électronique est une application fondamentale de la cryptographie. Elle a été découverte avec la cryptographie à clé publique par Diffie-Hellman.

Un schéma de signature permet à un utilisateur de signer des messages afin que la signature puisse être vérifiée ultérieurement. Où le destinataire pourra vérifier leur authenticité, leur origine, mais également de s'assurer qu'ils sont intacts. Ainsi, les signatures numériques à clé publique garantissent l'*authentification* et l'*intégrité* des données. Elles fournissent également une fonctionnalité de *non répudiation*, afin d'éviter que l'expéditeur ne prétende qu'il n'a pas envoyé les informations. Ces fonctions jouent un rôle tout aussi important pour la cryptographie que la confidentialité.

Une signature numérique a la même utilité qu'une signature manuscrite. Cependant, une signature manuscrite peut être facilement imitée, alors qu'une signature numérique est pratiquement infalsifiable. De plus, elle atteste du contenu des informations, ainsi que de l'identification du signataire.

3.1. Principe général

Pour atteindre ces objectifs, au lieu de crypter un message avec la clé publique du destinataire, le signataire doit les crypter avec sa propre clé privée. Et si des messages peuvent être décryptés avec la clé publique d'une personne bien définie, c'est que c'est cette personne qui les a vraiment créés.

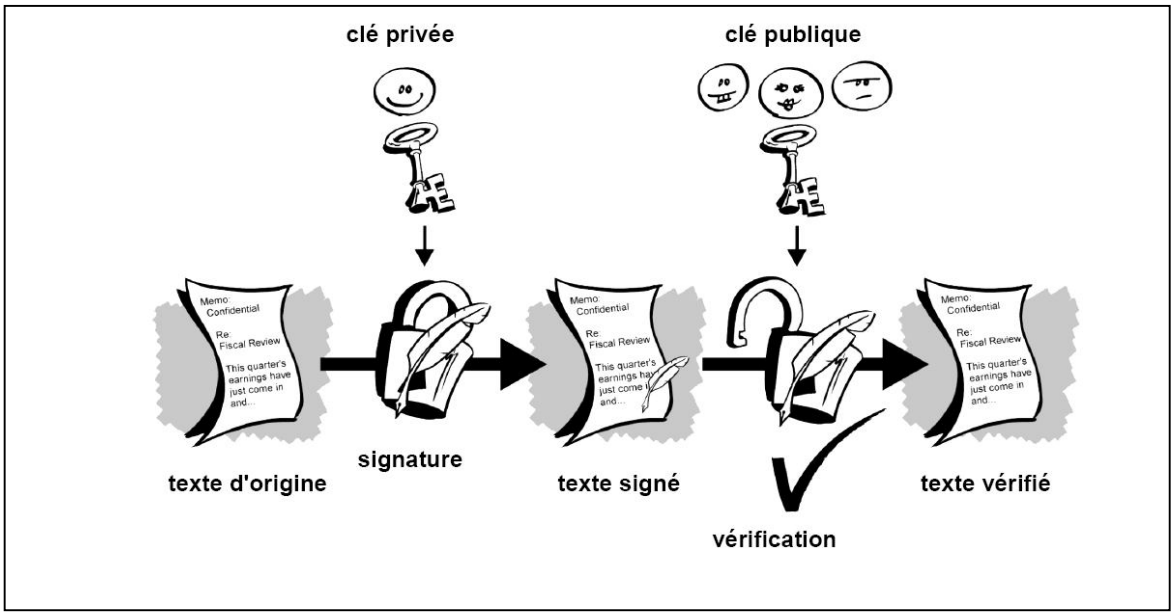


Figure 11 : ***Schéma de signature numérique simple***

Un tel schéma de signature est dit signature simple : un signataire signe des informations avec sa clé privée, il prouve l'authenticité et la non-répudiation des informations.

3.2. **L'utilisation de la fonction de hachage :**

Le système décrit précédemment comporte certains problèmes. Il est lent et produit un volume important de données (au moins le double de la taille des informations d'origine). L'ajout d'une *fonction de hachage* à sens unique dans le processus permet d'améliorer le schéma ci-dessus. Cette fonction traite une entrée de longueur variable (dans ce cas, un message pouvant contenir indifféremment des milliers ou des millions de bits), afin d'obtenir en sortie un élément de longueur fixe, à savoir 160 bits. En cas de modification des données (même d'un seul bit), la fonction de hachage garantit la production d'une valeur de sortie complètement différente.

Avant de signer son message, le signataire, doit utiliser une fonction de hachage qui génère un élément de données de longueur fixe, appelé *le condensé*. En outre, toute modification apportée aux messages entraine un résumé complètement différent.

Ensuite, il peut utiliser sa clé privée pour signer le condensé, il le joigne à son texte clair original et l'envoi sur un canal de transmission. A la réception du message, le destinataire peut procéder à la vérification. Il déchiffre de condensé avec la clé publique du signataire, hache le message original avec la même fonction de hachage et compare les deux : le haché obtenu avec le haché reçu. La signature est vérifiée si les deux hachés sont égaux.

Si la fonction de hachage utilisée est sécurisée, il est impossible de récupérer la signature d'un document pour la joindre à un autre document ou d'altérer un message signé. Ainsi, la moindre modification apportée à un document signé entraine l'échec du processus de vérification.

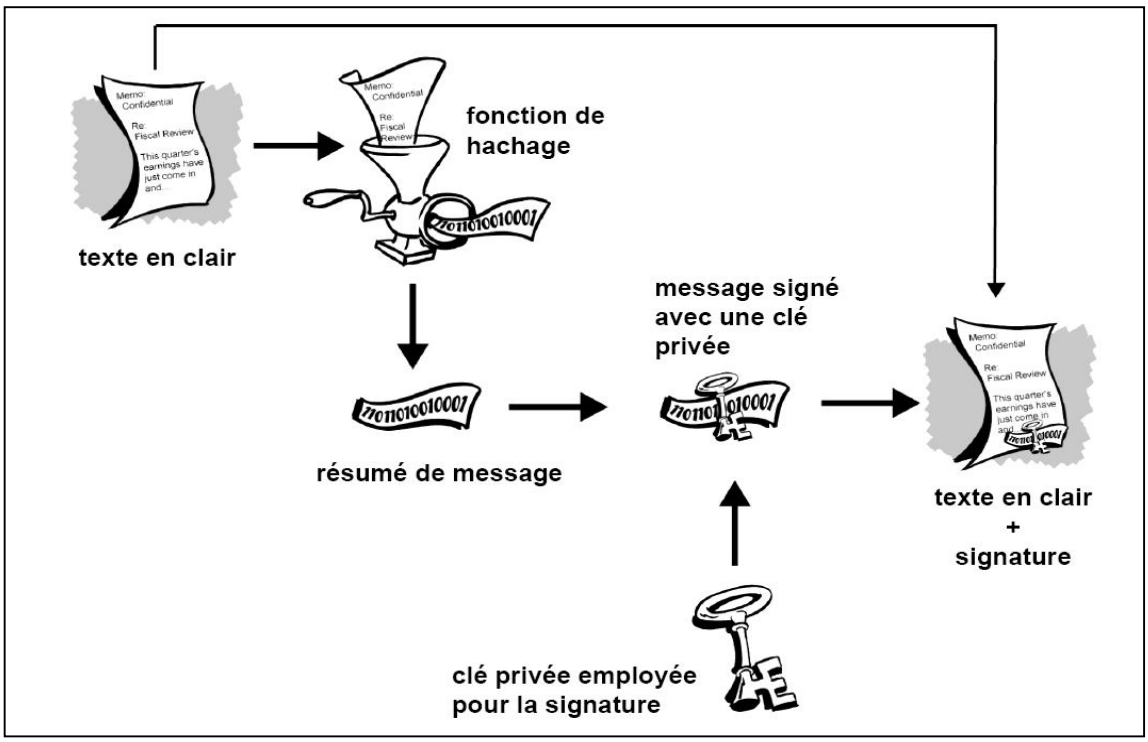


Figure 12 : ***signature numérique avec fonction de hachage***

Dans ce cas, un autre critère de sécurité est satisfait : l'intégrité de données.

3.3. Les attaques contre la signature numérique

La propriété mise en jeu dans un schéma de signature est la non-répudiation, qui est l'impossibilité pour un signataire de renier ses propres signatures ; ce qui signifie qu'une signature engage son auteur. Cette propriété est caractérisée par l'infalsifiabilité (en anglais : unforgeability).

La falsification est un message accompagné de sa signature (valide), mais qui a été produit par une personne autre que le réel signataire.

Si une telle falsification est impossible, cela signifie que le signataire (le réel possesseur de la clé privée) est le seul à pouvoir produire la signature d'un message donné. La non-répudiation est donc assurée.

3.3.1. Les différents types d'attaques

Le but d'un adversaire est de casser une signature, de façon qu'il puisse produire une signature valide qui sera acceptée par d'autres personnes. Nous décrivons ici les trois types d'attaques les plus couramment utilisées contre les schémas de signature.

- **Attaque sans message**

Dite aussi attaque à clé publique connue. Dans ce type d'attaque, la seule donnée à la disposition de l'attaquant est la clé publique du signataire. Ce type d'attaque est le plus faible que l'on puisse considérer.

- **Attaque à message**

Ici, l'adversaire est capable d'examiner des signatures soit à messages connus ou à messages choisis. Ce type d'attaque peut être subdivisé en deux classes principales :

- ***Attaque à message connu :*** dans ce type d'attaque, l'adversaire dispose de la clé publique du signataire et une liste de paires de messages/signatures.

Cette liste de message n'est pas choisie par l'adversaire.

- ***Attaque à message choisi*** : dans ce type d'attaque, l'adversaire peut demander à un signataire de lui signer un ensemble de messages choisis par lui-même.

Cette attaque peut être subdivisée elle aussi en deux sous types : attaque générique et attaque orientée :

- Pour l'attaque générique, l'adversaire choisie la liste indépendamment de la clé publique du signataire (la liste sera la même pour toutes les clés attaquées).

- Et l'attaque orientée, où le choix de la liste est fait après avoir eu la clé publique du signataire (l'attaque est orientée contre un signataire spécifique).

3.3.2. La sécurité des schémas de signature

Un schéma de signature est considéré est sûr, si aucun attaquant ne pourra produire une falsification.

On peut distinguer différents niveaux de cryptanalyse pour un adversaire :

- **Cassage total**

Ici l'adversaire est capable de calculer la clé privée du signataire ou de trouver un algorithme de signature aussi efficace, qu'il puisse produire des signatures valide. Dans ce cas, l'adversaire peut, en toute circonstance, agir comme le vrai signataire.

C'est l'attaque la plus dangereuse et la plus difficile à accomplir.

- **Falsification universelle**

Ici, l'adversaire ne cherche pas directement la clé secrète, mais par autres moyens, il peut construire un algorithme de création de signatures valides. Ceci est, en pratique, équivalent au cassage total du schéma.

- **Falsification sélective (existentielle)**

L'adversaire est capable de produire une signature à un message dont le signataire légitime n'a pas encore produit de signature.

Il existe un sous type de la falsification existentielle, où, l'adversaire est capable de générer de nouvelles paires message/signature dite : *falsification existentielle forte*

3.4. Le certificat numérique

Un problème qui subsiste toujours dans le domaine de signature numérique est la validité des clés.

Que se soit pour la signataire ou le vérificateur, tous les deux ont besoin d'une assurance que la paire de clés est valide pour une période bien définie.

Afin de réaliser cette assurance, une nouvelle entité nécessaire dans ce processus et *l'autorité de certification.* Cette tierce partie est responsable de l'identification des personnes ayant besoin d'une paire de clé (clé privée/clé publique). Alors elle délivre un fichier numérique qui permet de lier une clé aux données propres à une personne ou une entité (ces données peuvent être le nom ou l'adresse de cette personne). Ce fichier est appelé : *certificat numérique (électronique).* Un certificat doit être signé par l'autorité de certification qui le certifie.

Un certificat électronique est comparé physiquement à une carte d'identité d'une personne dans le but de certifier l'identité de la personne qui a signé un document électronique. Donc un certificat peut contenir : la clé publique du titulaire, son nom, la duré de validité de la clé, ainsi que des informations concernant l'autorité de certification (son nom et sa signature).

Un certificat électronique repose sur deux concepts : l'autorité de certification (AC) et l'infrastructure de gestion des clés (IGC).

3.4.1. L'autorité de certification (AC) :

Une autorité de certification a pour but principal, la validation d'un certificat délivré. Ceci permet de s'assurer que le certificat n'est pas modifié.

Le problème parvient si une clé privée est volée, le certificat associé ne sera plus digne de confiance. Dans ce cas, il faut avertir les utilisateurs potentiels du certificat qu'il n'est plus valide. Et l'autorité de certification doit mettre à jour la liste de révocation des certificats compromis : annonce de l'invalidité du certificat à partir de la date de déclaration de la perte de la clé privée.

La révocation de certificat n'empêche pas la vérification de signatures établies avant la date de révocation du certificat (un certificat invalide ne doit pas interdire la vérification).

3.4.2. L'infrastructure de gestion des clés (IGC) :

Une infrastructure de gestion des clés doit assurer plusieurs fonctionnalités :

- Elle vérifie l'identité du titulaire lors de l'émission d'un certificat.
- Elle se charge de la publication du certificat.
- Elle assure le renouvellement à la fin de validité du certificat.
- Elle gère la révocation en cas de perte de la clé privée.

3.5. Quelques exemples de signatures électroniques

Comme il existe plusieurs schémas de cryptosystèmes, il y a aussi plusieurs autres schémas de signatures qui diffèrent l'un de l'autre dans l'un des algorithmes, que se soit celui de la génération de la paire de clés, de signature, de vérification ou du coté de la signature. D'après ces différences on distingue certains algorithmes de signature numérique.

3.5.1. La signature numérique RSA

RSA est le plus célèbre et le plus utilisé des algorithmes cryptographiques à clé publique depuis sa publication en 1978. Il peut être utilisé pour le cryptage ainsi que pour la signature. Sa sécurité se base sur la difficulté de factorisation des grands nombres entiers.

L'idée est de calculer un module N comme le produit des deux nombres premiers p et q et l'exposant e co-premier avec $\varphi(N)$.

Le schéma de signature RSA peut être décrit comme suit :

- **La génération de clés** : un générateur de clé RSA, prend comme paramètre la longueur de clé K et choisie aléatoirement deux nombres premiers distincts p et q dont leur produit donne le module :

$$N = pq.$$

Ensuite, il choisit (aléatoirement aussi) un autre nombre e ϵ [1.. $\varphi(N)$] qui sera l'exposant de cryptage et calcule son exposant correspondant d, l'exposant de décryptage tel que :

$$ed = 1 \bmod \varphi(N).$$

En sortie on aura : (N, e, d) où :

(N, e) : est la clé publique

(N, d) : est la clé privée.

- **La signature** : ressemble beaucoup au procédé de cryptage mais cette fois, on utilise la clé privée pour signer le message *m*: le signataire calcule alors :

$$s = m^d \pmod N.$$

- **La vérification** : pour vérifier une signature, le vérificateur a besoin de la clé publique (N,e) pour calculer :

$$m = s^e \bmod N.$$

Un tel schéma ne peut pas être utilisé seul en pratique, il ne pourra pas résister aux attaques à message choisi. Un adversaire qui veut obtenir une signature sur un message m doit trouver deux messages m_1 et m_2 tel que :

$$m = m_1 m_2 \ (mod\ N),$$

il demande au signataire une signature sur m_1 puis sur m_2 puis il calcule m :

$$s(m) = m^d \pmod N = (m_1^d\ m_2^d) \pmod N = s(m_1)\ s(m_2) \pmod N$$

La solution à ce problème est d'utiliser une fonction de hachage *h* sur le message avant de le signer. Donc le schéma de signature change partiellement dans les deux dernières phases :

- **La signature** : le signataire doit d'abord calculer le condensé du message h(m) puis il le signe :

$$s = (h(m))^d \bmod N.$$

- **La vérification** : Le vérificateur vérifie la signature du condensé :

$$h(m) = s^e \bmod N.$$

3.5.2. La signature digitale DSA (Digital Signature Algorithme)

La signature digitale DSA est l'une des techniques de signature les plus utilisée où la fonction de hachage est un outil indispensable ainsi que certains nouveaux paramètres dont on peut citer :

p : un module premier, où $2^{L-1}<p<2^{L}$ et L est la longueur en bits de p.

q : un nombre premier diviseur de (p-1), où $2^{N-1}<q<2^{N}$ et N est la longueur en bits de q.

g : un générateur de sous-groupe d'ordre q mod p tel que $1< g < p$.

x : la clé privée qui doit rester secrète ; x est un entier généré aléatoirement, tel que $0<x <q$ ($x\epsilon[1..q-1]$).

y : la clé publique, où $y = g^{x} \bmod p$.

k : un entier choisi aléatoirement tel que $0 < k < q$. k est secret et unique pour chaque message.

k^{-1} : l'inverse multiplicatif de k modulo q. $k^{-1} \epsilon[1..q-1]$ et $k\, k^{-1} \equiv 1 \bmod q$.

p , q et g sont dits les paramètres du domaine.

- ***Génération de signature DSA***

La signature d'un message M consiste aux deux nombres r et s calculés comme suit :

$r = (g^{k} \bmod p) \bmod q$.

z = les N premiers bits les plus à gauche du condensé h(M).

$s = (k^{-1} (z + xr)) \bmod q$.

Pour calculer s, le condensé doit être convertis en entier.

Les valeurs de r et s doivent être vérifiées afin de déterminer si r = 0 ou s = 0. Si = 0 ou s = 0, une nouvelle valeur de k doit être recalculée. Il n'est pas souhaitable d'avoir = 0 ou s= 0 si la signature est calculée proprement.

Maintenant, la signature (r , s) peut être transmise avec le message au vérificateur.

- ***Vérification et validation de la signature DSA***

La vérification de la signature peut être réalisée par n'importe quelle entité (le signataire lui-même, le récepteur légitime, ou un adversaire (intercepteur de la signature)) utilisant la clé publique du signataire. Le signataire voudrait vérifier que sa signature est correcte avant sa transmission, et le récepteur (ou autre personne) voudra vérifier l'authenticité de la signature.

Avant de procéder à la vérification, les paramètres de domaine p, q et g et la clé publique du signataire doivent être obtenus d'une manière authentifiée comme par exemple depuis un certificat numérique (donné par une autorité de certification).

Soit :

M', r' et s' : les versions reçues de M, r et s,

y : la clé publique du signataire,

N : la longueur de q, donnée en nombre de bits.

1. Le vérificateur doit vérifier que $0 < r' < q$ et $0 < s' < q$. si une des conditions des n'est pas vérifiée, la signature sera rejetée comme invalide ;

2. Si les deux conditions précédentes sont vérifiées, le vérificateur calcule ce qui suit :

$w = (s')^{-1} \bmod q$, où $(s')^{-1}$ est l'inverse multiplicatif de s' modulo q,

z = les N premiers bits les plus à gauche du condensé h(M'),

$u1 = (zw) \bmod q$,

$u2 = ((r')w) \bmod q$,

$v = (((g)^{u1} (y)^{u2}) \bmod p) \bmod q$.

3. Si v = r', alors la signature est vérifiée,

4. Si $v \neq r'$, alors soit le message soit la signature a été altéré (il y a eu une erreur dans le processus de génération de signature ou alors une troisième personne a tenté de falsifier la signature). La signature est considérée comme invalide.

Les exemples de signature électronique ne se restreignent pas dans les deux schémas cités mais s'étendent et se distinguent sur plusieurs domaines d'application, en satisfaisant plusieurs critères de sécurité comme la confidentialité, l'intégrité et l'anonymat qui est l'objectif principal de ce qui vas suivre dans ce mémoire.

Chapitre 4 :

ANONYMAT ET SIGNATURES ANONYMES

Face aux nombreuses menaces contre la vie privée qui existent du fait de l'utilisation d'internet (profiling, traces électroniques,…), plusieurs solutions techniques tendant à préserver l'identité invisible des utilisateurs, en leur permettant de ne pas dévoiler leur véritable identité, ont été développées. Certaines de ces techniques font objet de notre travail.

4.1. Terminologie

Le problème de l'anonymat ne réside pas en un seul terme, mais il se distingue en plusieurs notions. Afin d'aboutir à une définition exacte (ou proche du but) il faut passer par certaines expositions qui commencent par la protection de la vie privée (confidentialité) des individus, l'identité des personnes, ainsi que l'utilisation d'autres concepts dérivants.

Cette section est consacrée principalement, aux concepts du monde de l'anonymat.

4.1.1. La protection de la vie privée (confidentialité, en anglais : privacy)

Le concept de protection de la vie privée est une notion très vaste. Et comme il est perçu différemment, il est ouvert à l'interprétation indépendamment des individus et du type de société.

La confidentialité (la protection de la vie privée) peut être définie comme la délivrance de toutes intrusions non autorisées, ou le droit de garder les affaires personnelles et les relations de l'individu secrètes.

Cette notion n'est pas seulement large, mais elle peut aussi apparaître sous différentes dimensions, tel que la confidentialité des personnes, la confidentialité du comportement personnel, la confidentialité des communications et la confidentialité des données :

- La confidentialité des personnes se réfère à l'intégrité du corps de l'individu,
- La confidentialité du comportement personnel concerne le comportement de l'individu, en particulier, ses attitudes socioculturelles dans les lieus privés ou publiques,
- La confidentialité des communications repose sur le but de communiquer avec d'autres entités sans qu'il y ait le contrôle habituel des communications par d'autres individus ou organisations,
- La confidentialité des données personnelles concernant l'objectif d'interdire autres personne ou organisation de manipuler les données personnelles d'un individu.

4.1.2. L'identité numérique

La notion d'identité devient très importante avec l'émergence des sciences. Principalement, on peut mentionner deux différents types d'identité : identité physique et identité numérique :

- L’identité physique désigne les caractéristiques ou la personnalité d’un individu. Alors, l’identité ne réside pas en un nom seulement, mais aussi en un ensemble de caractéristiques décrivant une personne. Ces caractéristiques peuvent inclure, entre autres, les caractéristiques physiques inchangeables d’une personne, ces préférences, ou les perceptions d’autres personnes sur la personnalité d’un individu.

- L’identité numérique est un terme assez difficile à définir. A un niveau général, l’identité numérique peut être définie comme une représentation lisible par une machine d’une identité humaine utilisée dans les systèmes électroniques permettant d’interagir avec d’autres machines ou individus.

Le but de l’identité numérique est de permettre la fonctionnalité du contrôle d’accès ou de lier une transaction particulière ou un ensemble de données dans un système informatique à un individu identifiable.

A l’aide de l’identité numérique, un individu peut être identifié, authentifié, et autorisé d’accéder à un service ou une ressource donnée. La sécurité des systèmes informatiques repose sur le pouvoir d’identification et l’authentification des utilisateurs.

Selon l’organisation ou la personne à qui un individu communique ses informations personnelles, il a un contrôle partiel sur les données qui constituent sont identité numérique comme il peut choisir la quantité d’information à divulguer à un fournisseur de service.

A partir de cette quantité de données que l’individu peut données à une autre personne ou à une organisation sur son identité, l’identité numérique peut être anonyme, pseudonyme, identifiable ou identité de groupe. Une identité avec identification totale, permet à un fournisseur de service de lier une transaction à un individu. Une transaction pseudonyme peut être liée à un pseudonyme mais pas à un individu identifiable. Une transaction impliquant une identité anonyme ne pourra jamais être liée à un individu identifiable. En fin pour l’identité du groupe, une transaction est liée à un groupe d’utilisateur, mais il est impossible de la lier à un membre bien précis de ce groupe.

4.1.3. L’anonymat

Le concept d’anonymat peut être défini comme « l’état de ne pas être connu, non nommé, ou non identifié ». Dans le domaine des systèmes d’information, l’anonymat peut être défini par rapport à un ensemble d’individus, aux entités communicantes ou à l’évidence résultante d’une transaction.

Détailler cette définition, mène à étudier les différentes notions en rapport avec l’anonymat.

- **L’anonymat au milieu d’un groupe**

Si on prend la définition de Pfitzmann où l’anonymat est défini par « l’état d’être non identifiable au milieu d’un ensemble d’individus, l’ensemble d’anonymat ». L’ensemble d’anonymat est l’ensemble de tous les sujets pouvant causer une action, par exemple, l’ensemble de tous les utilisateurs d’un système.

Cette définition exprime brièvement le but principal de l'anonymat, la question d'être identifiable ou non. L'anonymat est la propriété fondamentale de cacher l'identité, procurant une totale élimination des informations pouvant identifier un individu. Toute information d'identification exigée par une entité communicante pourra être ajoutée à un canal de données sur un système anonyme. L'anonymat procure, alors, la liberté du choix du taux d'informations d'identification révélées si nécessaire.

- **L'ensemble d'anonymat**

La méthode traditionnelle de mesurer l'anonymat est l'utilisation de la taille de l'ensemble d'anonymat qui est l'ensemble de tous les participants qui auraient pu exécuter une action. Cette approche a été utilisée pour analyser l'anonymat obtenu par les systèmes de communication anonyme, où : de plus la taille de l'ensemble des participants ayant pu exécuter une opération est large, le plus l'anonymat obtenu par un système est fort.

Cette technique de mesure de la force de l'anonymat, la plus utilisée, n'est pas idéale. L'inconvénient majeur de l'ensemble d'anonymat est qu'il suppose une distribution uniforme de probabilité entre les participants de l'ensemble. Cette hypothèse est plutôt injuste dans un groupe d'utilisateurs hétérogène. En réponse à cette faiblesse, un certain nombre de méthode de mesure de d'anonymat a été proposé pour gérer les différents problèmes rencontrés avec l'ensemble d'anonymat.

- **L'anonymat des entités communicantes.**

Dans un système de communication traditionnelle, il doit y avoir deux participants : un émetteur et un récepteur, où l'un ou l'autre peut être un groupe d'individus. Alors, à qui l'anonymat doit être appliqué ?

L'anonymat vu du coté système de communication, peut être subdivisé en trois parties : anonymat de l'émetteur, anonymat du récepteur et anonymat des deux parties. L'anonymat de l'émetteur signifie qu'un observateur ne pourra jamais lier un message particulier à un émetteur précis (mais le récepteur peut être connu). L'anonymat du récepteur signifie qu'un observateur ne pourra pas lier un message particulier à un récepteur précis (l'émetteur peut être connu). En fin, il est peut être désirable que les deux entités communicantes soient anonymes afin de rester loin de toute observation et le message devient non chaînable.

- **L'anonymat des transactions**

Une transaction est souvent documentée par un enregistrement électronique. Une transaction anonyme est telle qui ne pourra pas être attribuée à un individu particulier et un enregistrement qui documente une transaction est anonyme s'il ne peut pas être associé à un sujet de transaction.

Dans ce cas l'anonymat de l'émetteur ne suffira pas pour assurer l'anonymat d'une transaction. Et le message lui-même ne devra pas contenir des informations qui lient l'évidence d'une transaction à un citoyen existant. Donc, un utilisateur doit rester anonyme à égard des communications et les enregistrements résultants des transactions.

4.1.4. Le pseudonymat

Un pseudonyme est un nom fictif ou une fausse identité que chaque individu peut choisir afin de l'utiliser comme un identificateur au lieu d'utiliser son vrai nom.

Le pseudonymat est la propriété d'utiliser un pseudonyme comme identificateur. Un individu peut avoir plusieurs pseudonymes afin d'utiliser un pseudonyme distinct pour accéder à chaque système pour éviter d'être lié aux données transactionnelles accumulées dans chaque système (une opération qui pourra servir dans l'identification de la personne).

Une transaction pseudonyme est celle qui peut être liée à un pseudonyme mais pas à un individu particulier. La différence entre l'anonymat et le pseudonymat est que le pseudonyme permet à un fournisseur de service de lier toutes les transactions impliquant le même pseudonyme.

Les pseudonymes peuvent être classés selon la façon dont ils étaient créés et leurs relations avec l'identité réelle de l'utilisateur :

- Pseudonyme personnellement généré : un pseudonyme généré par son propriétaire, est le seul à pouvoir établir le lien entre ce pseudonyme et l'identité réelle (ex : les pseudonymes utilisés pour le chat sur internet).

- Pseudonyme de référence : c'est un pseudonyme qui peut être transformé en une identité réelle à l'aide d'une liste de références qui est protégée par une autre entité dite troisième partie confiante « *trusted third party* » (numéro de sécurité sociale).

- Pseudonyme cryptographique : un pseudonyme généré en appliquant une fonction cryptographique sur des données d'identification. Cette fonction peut être à sens unique, ce qui donne un pseudonyme à sens unique.

4.1.5. La non chaînabilité (non-reliabilité)

La non chaînabilité représente l'impossibilité (pour d'autre utilisateurs) d'établir un lien entre différentes opérations réalisées par le même utilisateur, par exemple, en interdisant la fusion ou le croisement d'informations à partir de différentes fichiers ou bases de données.

4.1.6. Révocation d'anonymat

Dans les transactions anonymes ou pseudonymes, un individu ne divulgue pas son identité aux autres parties. Cela est indésirable car une personne anonyme ne pourra être responsabilisée de ces actions. Dans ce cas, un système anonyme doit être divisé : dans des circonstances normales, un individu peut communiquer d'une manière anonyme sans être obliger de divulguer son identité, mais dans des circonstances exceptionnelles, cette identité ne doit pas rester cachée non plus.

Afin d'accomplir une telle procédure, il faut introduire une troisième partie qui se chargera de dévoiler l'identité d'un individu sous certaines conditions spécifiques : c'est la notion de révocation d'anonymat. Le concept permettant la révocation d'anonymat ou du pseudonymat est dit : *séquestre d'identité* (identity escrow)

Dans un tel système à anonymat révocable, un utilisateur doit disposer d'un certificat qui lui est remis par une autorité de certification. Ce certificat permet à l'utilisateur de prouver ces droits d'accès aux différents services proposés, cette phase d'authentification est anonyme (l'anonymat est considéré par rapport à un groupe de personne possédant les droits d'accès à ce service). Et il doit y avoir une troisième entité

capable de lever cet anonymat si nécessaire appelée *agent de séquestre* (l'agent séquestre peut être différent de l'autorité de certification).

Un schéma de séquestre d'identité IdE est défini par quatre algorithmes :

- IdE.KGen : un algorithme de génération de clés, il calcule les paramètres publics du système pk et donne les clés secrètes sk_A et sk_S respectivement à l'autorité et à l'agent de séquestre.

- IdE.Reg « R(U ; A(sk_A))(pk) » un protocole d'émission de certificat . un protocole interactif entre un utilisateur U est l'autorité A permettant de délivrer un certificat C à l'utilisateur U.

- IdE.Verif « V(U(C) ;V)(pk) » un protocole de vérification. Un protocole interactif entre l'utilisateur U est un vérificateur V où l'utilisateur persuade le vérificateur qu'il connaît un certificat C délivré par l'autorité et lui donne un certificat de séquestre (escrow certificat).

- IdE.Open algorithme d'ouverture. L'agent de séquestre lève l'anonymat en retrouvant l'identité de l'utilisateur à partir du certificat de séquestre.

4.2. Signature de groupe

Les signatures numériques sont rapidement devenues omniprésentes dans presque tous les domaines de la vie électronique. Elles sont utilisées pour obtenir certains services de sécurité comme l'authentification, l'intégrité de données et la non répudiation. Les signatures de groupe représentent un nouveau concept introduit par Chaum et Van Heyst en 1991. Similaire à sa correspondante, la signature de groupe permet à un utilisateur de signer un document au nom d'un groupe d'utilisateurs lui permettant l'anonymat de son identité. Cet anonymat est absolu vis-à-vis d'un vérificateur externe ou un autre membre du groupe, sauf pour le gestionnaire (manager) du groupe, qui peut découvrir l'identité du signataire si nécessaire. Elle est aussi publiquement vérifiable en possédant la clé publique du groupe. La signature de groupe doit être conçue de façon que aucun membre du groupe ni le manager ne pourra falsifier la signature d'un autre membre du même groupe sur un document donné.

Le concept de signature de groupe a été introduit et réalisé par Chaum et Van Heist en 1991, où ils proposaient quatre schémas de signature : trois d'entre eux apportent un anonymat calculatoire tandis que le quatrième fournit un anonymat théorique. Malheureusement, certains schémas ne permettent pas au manager d'ajouter de nouveaux membres après la création du groupe et d'autres nécessitent que le manager contacte tous les membres afin de pouvoir ouvrir une signature (afin de pouvoir révéler l'identité d'un signataire).

En 1994 , Chen et Pedersen [CP94] ont proposé leur schéma permettant d'introduire une nouvelle entité indépendante du groupe qui pourra ouvrir la signature . Ensuite Peterson proposa en 97 [Pet97]. la première construction générique de signature de groupe à partir de n'importe quel schéma de signature électronique.

Un autre schéma publié par Camenisch et Stadler en 1997 [CS97] qui repose sur l'indépendance des clés et des signatures du nombre de membres dans le groupe. Un schéma

plus efficace résistant aux coalitions face à des attaques adaptatives réalisé par Ateniese, Camenisch, Stadler et Tsudik en 2000 [ACJT00] repose sur les hypothèses RSA fort et Diffie-Hellman décisionnel.

Les recherches ensuite se focalisent sur le problème de réduire la taille de signature et augmenter l'efficacité et la sureté des schémas. Bellar, Micciancio et Warinschi, [BMW03] alors, en 2003, proposaient leur résultat sur les signatures de groupes statiques. En 2005, Zhang, Wu et Wang ont publié leur résultat [ZWW05] qui présente un nouveau schéma de signature de groupe plus efficace avec des paramètres de sécurité plus avancés. La construction de Boneh, Boyen et Shacham [BBS04] en 2004, par la suite, a fourni une signature courte (de la taille d'une signature RSA) mais avec un niveau de sécurité plus élevé basée sur l'hypothèse RSA fort ainsi qu'une nouvelle hypothèse , appelée l'hypothèse linéaire décisionnelle.

Basé sur les hypothèses de systèmes pseudonymes, Camenisch et Lysyanskaya présentent en 2004 un nouveau schéma [CL04]. Par la suite, Bellare, Shi, et Zhang ont mis un nouveau modèle de sécurité pour les groupes dynamiques avec une construction générique de schéma de signature de groupe [BSZ05].

4.2.1. Fonctionnement et construction

Une signature de groupe fonctionne comme suit : un manager de groupe met en place un groupe et lui crée une clé publique qui va être commune à tous les membres du groupe.

Quand un membre se joint au groupe il obtient une clé secrète d'adhésion au groupe à l'aide d'un secret d'adhésion. Si une révocation d'anonymat est désirée, un secret additionnel est utilisé dans le but de découvrir l'identité du signataire. Maintenant, le membre peut générer une signature au nom du groupe et le vérificateur peut déterminer si le signataire est membre du groupe, mais ne pourra avoir aucune information sur son identité.

Dans certains cas spéciaux, comme le cas d'opposition légale (un conflit), le manager du groupe peut décider d'ouvrir la signature et divulguer l'identité du signataire.

Alternativement, cette procédure peut être attribuée à une tierce partie, appelé : manager d'anonymat. Le manager d'anonymat peut examiner, si nécessaire, cette signature et aide à découvrir l'identité du membre qui a engendré cette signature.

Donc, les entités participantes dans un schéma de signature de groupe sont les suivantes :

- **Manager de groupe** : le manager du groupe est le responsable du fondement du groupe et ses paramètres : il est chargé du choix de clés du groupe, d'adhésion des nouveaux membres et si nécessaire, il est chargé de la révocation d'anonymat d'un signataire (si cette fonctionnalité n'est pas attribuée au manager d'anonymat).

- **Membre du groupe ou utilisateur** : c'est la personne souhaitant joindre le groupe, s'inscrit au niveau du manager et peut ensuite engendrer des signatures anonymement.

- **Manager d'anonymat (optionnel)** : une tierce confiante qui peut être séparée du manager du groupe et qui se charge de la révocation d'anonymat.

- **Vérificateur** : un vérificateur est la personne ayant reçue une signature et souhaitant vérifier sa provenance. Un vérificateur peut seulement savoir si le signataire est réellement membre du groupe sans avoir la moindre information sur son identité.

Un schéma de signature de groupe comprend les protocoles suivants :

- ✓ **Setup :** un algorithme de génération de paramètres du groupe. Il prend en entrée un paramètre de sécurité *k* et génère une clé publique du groupe pk_G , une clé secrète du manager du groupe sk_{MG} , une clé secrète du manager d'anonymat (autorité de révocation) sk_{RA} , et des clés secrètes pour chaque utilisateur sk_U .

- ✓ **Join** : un protocole interactif entre le manager du groupe et les nouveaux utilisateurs. Il est exécuté à chaque fois qu'un utilisateur souhaite joindre le groupe, et la fin de cette procédure, le nouveau membre aura son certificat d'appartenance au groupe ainsi que sa clé secrète de signature sk_U .

- ✓ **Sign** : un algorithme de signature, qui prend en entrée un message m, la clé secrète du l'utilisateur sk_U et retourne une signature s du message.

- ✓ **Verify :** un algorithme de vérification de signature, qui utilise le message m, la signature s et la clé publique du groupe pk_G pour remettre au vérificateur la valeur 1 si la signature est valide et 0 sinon.

- ✓ **Open** : un protocole entre le vérificateur (qui doit être une entité de confiance) et l'autorité de révocation (qui peut être le manager du groupe ou le manager d'anonymat) permettant l'ouverture de la signature. Il prend en entrée le message m, la signature s, la clé publique du groupe pk_G et la clé secrète de l'autorité de révocation sk_{RA} et donne en sortie l'identité U du signataire.

Cette interprétation présente les différents algorithmes nécessaires pour la construction d'un schéma de signature de groupe. Certains schémas ne comprennent pas la phase Join, ce qui ne permet pas l'intégration de nouveaux membres. On considère alors deux types de groupes :

- Les groupes statiques, dans lesquels la taille du groupe est fixé au départ de la création du groupe et le groupe n'acceptera ni l'adhésion de nouveaux membres ni la suppression des membres existants.
- Les groupes dynamiques, dans lesquels, les membres peuvent soit adhérer au groupe soit le quitter à n'importe quel moment.

4.2.2. Notions de sécurité existantes :

Tout schéma de signature doit satisfaire les propriétés suivantes :

- **L'inforgeabilité (la non-falsification) :** un besoin basic de tous les schémas de signature est que les signatures soient inforgeables, i.e. : il est calculatoirement infaisable de produire plusieurs paires message/signature (m/s) qui soient acceptées par l'algorithme de vérification sans avoir connaissance de la clé secrète.

Alors l'inforgeabilité est que seuls les membres d'un groupe peuvent signer au nom de leur groupe.

- **L'exactitude :** toute signature valide produite par un membre du groupe utilisant l'algorithme Sign doit être acceptée par l'algorithme Verify.

- **L'anonymat :** ayant une signature de groupe valide, il est strictement impossible à toute personne sauf l'autorité de révocation d'identifier le signataire actuel.

- **La non-reliabilité** : décider si deux différentes signatures valides proviennent d'un même signataire est calculatoirement infaisable.

- **La non-diffamation** : ni les membres d'un groupe ni même le manager ne peuvent produire une signature valide au nom d'un autre membre.

- **La traçabilité** : l'autorité de révocation doit être toujours capable de lever l'anonymat d'un signataire à partir d'une signature valide.

- **La résistance aux coalitions** : aucune coalition de membres ne peut produire de signature qui ne soit pas ouvrable sur l'identité d'un des membres de la coalition.

4.2.3. Utilisation des signatures de groupe :

Comme technique de signature qui préserve l'anonymat du signataire, la signature de groupe a été proposée par Chaum et Van Heist dans une application assez particulière dans une entreprise, où les employés peuvent utiliser une imprimante anonymement. En cas d'abus, l'anonymat de l'employé peut être levé. Mais son utilisation ne s'est pas arrêtée dans cette application, et plusieurs autres cas ont été proposés, soit en utilisant directement la signature de groupe soit une de ces variantes. Par exemple : dans les enchères en ligne. Les enchérisseurs s'enregistrent au prés d'une autorité d'anonymat ; ils obtiennent un certificat leur permettant d'enchérir anonymement, et seule l'anonymat du meilleur enchérisseur est levé par l'autorité.

Une autre application est le vote électronique. Mais, comme l'anonymat dans les groupes est révocable, ce qui est contradictoire au principe des votes (garantie de l'anonymat des bulletins) une variante des signatures de groupe a été proposée par Canard et Traoré en 2005 [CSST05] satisfaisant les contraintes d'un schéma de vote électronique sûr et anonyme.

4.2.4. Schémas de signature de groupe

Cette partie présente certains modèles de signature de groupe proposés en expliquant les différents algorithmes qui constituent le schéma ainsi que les paramètres de sécurité assurés.

4.2.4.1. Le modèle ACJT

Ce modèle nommé ainsi des noms de ces inventeurs, dont la sécurité est basée sur l'hypothèse RSA fort et l'hypothèse Diffie-Hellman Décisionnelle, présente un schéma efficace de signature de groupe.

- **Le problème RSA Fort** : Soit n = pq le module du RSA et soit G un sous groupe cyclique Z . Étant données n et z ϵ G, le problème RSA Fort consiste à trouver u ϵ G et e ϵ Z satisfaisant $z \equiv u^e \bmod n$.

- **Le problème Diffie-Hellman décisionnel** : Soit G = < g > un groupe cyclique généré par g. Etant données g, g^x, g^y et g^z ϵ G, Le problème Diffie-Hellman décisionnel consiste à décider si les éléments g^{xy} et g^z sont égaux.

❖ **Les algorithmes du modèle :** ce modèle contient les différents algorithmes déjà cités pour un schéma de signature de groupe.

➢ **La création du groupe (Setup)**

Les différents paramètres nécessaires au système sont donnés par :

- $\varepsilon > 1$, k et l_p les paramètres de sécurité.

- $\lambda 1, \lambda 2, \gamma 1, \gamma 2$ des longueurs telles que $\lambda 1 > \varepsilon(\lambda 2+k)+2$, $\lambda 2 > 4l_p$, $\gamma 1 > \varepsilon(\gamma 2+k)+2$ et $\gamma 2 > \lambda 1 + 2$,

- les intervalles $\Lambda =]2^{\lambda 1} - 2^{\lambda 2}, 2^{\lambda 1} + 2^{\lambda 2}[$ et $\Gamma =]2^{\gamma 1} - 2^{\gamma 2}, 2^{\gamma 1} + 2^{\gamma 2}[$,

- H une fonction de hachage résistante aux collisions $H : \{0,1\}^* \rightarrow \{0,1\}^k$

Etant donnés ces paramètres, le manager du groupe :

i. Choisi aléatoirement deux nombres premiers secrets p′ et q′ de taille l_p , tels que p = 2p′ + 1 et q = 2q′ + 1 soient premiers. Il calcule le module n = pq.

ii. Choisi les éléments aléatoires a, a_0 , g et h ϵ_R QR(n) d'ordre p′q′

iii. Choisi un secret aléatoire x ϵ_R $Z_{p'q'}$ et pose $y = g^x \bmod n$

La clé secrète du manager est sk_{MG} = (p′, q′, x) et la clé publique du groupe est pkG=(n, a, a_0, y, g, h).

➢ **L'enregistrement d'un utilisateur (Join)**

Un nouvel utilisateur doit s'enregistré auprès du manager du groupe à l'aide du protocole Join.

Le protocole Join se déroule comme suit :

i. L'utilisateur U choisit une valeur secrète $\tilde{x}_i$ et envoie au manager MG un engagement sur x_i en prouvant que celui-ci est bien construit dans la preuve pok_1.

ii. Le manager renvoie deux aléas à l'utilisateur et celui-ci construit son secret x_i et envoie à MG la valeur $C_2 = a^{xi} \bmod n$ avec une preuve de consistance pok_2.

iii. Ce dernier peut maintenant calculer une signature de cet engagement et renvoie a U son certificat.

Lors de ces échanges, U produit deux preuves de connaissance :

1. dans pok_1, U prouve la connaissance de la représentation de C_1 en bases g et h.

   $pok_1 = pok(\mu, \nu : C_1 = g^{\mu}h^{\nu})$.

2. dans pok_2, U prouve a GM que :

   - le logarithme discret de C_2 en base a est dans l'intervalle Λ,
   - il connait les entiers u, v et w tels que :
     - u appartient a l'intervalle $] - 2\lambda 2 , 2\lambda 2 [$,
     - u est égal au logarithme discret de $C_2/a^{2\lambda 1}$ en base a,
     - $C_1^{\alpha i} g^{\beta i}$ est égal à $g^u(g^{2\lambda 2})^v h^w$.

   $pok2 = pok(u, v,w : u \epsilon] - 2^{\lambda 2} , 2^{\lambda 2} [$ et $a^u = C_2/a^{2\lambda 1}$ et $C_1^{\alpha i} g^{\beta i} = g^u(g^{2\lambda 2})^v h^w$).

A l'issue de l'exécution de ce protocole, l'utilisateur a obtenu un certificat d'appartenance au groupe (le couple $[A_i, e_i]$), ainsi qu'un secret connu de lui seul, qui lui permettra de signer anonymement. Et le manager ajoute le triple $\{ A_i, e_i, trans_i\}$ dans une liste de membre enregistrés.

- **La signature d'un message (Sign)**

Avec son certificat, le membre déjà inscrit peut signer des messages de son choix $m \epsilon \{0,1\}^*$. Pour ce faire, il exécute le protocole suivant :

i. U génère une valeur aléatoire $w \epsilon_R \{0,1\}^{2lp}$ et calcule :

   $T_1 = A_i y^w \bmod n$, $T_2 = g^w \bmod n$, $T_3 = g^{ei} h^w \bmod n$.

ii. U choisit $r_1 \in_R \pm\{0, 1\}^{\varepsilon(\gamma 2+k)}$, $r_2 \in_R \pm\{0, 1\}^{\varepsilon(\lambda 2+k)}$, $r_3 \in_R \pm\{0, 1\}^{\varepsilon(\gamma 1+2lp+k+1)}$ et $r_4 \in_R \pm\{0,1\}^{\varepsilon(2lp+k)}$ et calcule :

   (a) $d_1 = T^{r1}{}_1 /(a^{r2} y^{r3}) \bmod n$, $d_2 = T^{r1}{}_2 / g^{r3} \bmod n$, $d_3 = g^{r4} \bmod n$ et $d_4 = g^{r1} h^{r4} \bmod n$

   (b) $c = H(g||h||y||a_0||a||T_1||T_2||T_3||d_1||d_2||d_3||d_4||m)$;

   (c) $s_1 = r_1 - c(e_i - 2^{\gamma 1})$, $s_2 = r_2 - c(x_i - 2^{\lambda 1})$, $s_3 = r_3 - ce_i w$ et $s_4 = r_4 - cw$.

iii. U renvoie $(c, s_1, s_2, s_3, s_4, T_1, T_2, T_3)$.

La signature de connaissance que produit l'utilisateur peut être vue comme une signature de connaissance de son certificat.

- **La vérification d'une signature (Verify)**

Le destinataire vérifie la validité de la signature $(c, s_1, s_2, s_3, s_4, T_1, T_2, T_3)$ du message m de la manière suivante :

i. il calcule

$$c' = H(g||h||y||a_0||a||T_1||T_2||T_3||a^c{}_0 T_1{}^{s1-c2\gamma 1} /(a^{s2-c2\lambda 1 ys3}) \bmod n \,||T_2{}^{s1-c2\gamma 1} /g^{s3} \bmod n||T^c{}_2\, g^{s4} \bmod n||T_3{}^c\, g^{s1-c2\gamma 1} h^{s4} \bmod n||m)$$

ii. la signature est acceptée si et seulement si $c = c'$ et si $s_1 \in_R \pm\{0, 1\}^{\varepsilon(\gamma 2+k)+1}$, $s_2 \in_R \pm\{0, 1\}^{\varepsilon(\lambda 2+k)+1}$, $s_3 \in_R \pm\{0, 1\}^{\varepsilon(\gamma 1+2lp+k+1)+1}$ et $s_4 \in_R \pm\{0, 1\}^{\varepsilon(2lp+k)+1}$

- **Révocation d'anonymat (Open)**

En cas de conflit, le manager de groupe est capable de lever l'anonymat d'une signature.

1. MG vérifie la validité de la signature a l'aide du protocole Verify.
2. Il retrouve A_i en calculant $A_i = T_1/T^x{}_2 \bmod n$.
3. Il prouve que $\log_g y = \log_{T2}(T_1/A_i \bmod n)$.

❖ **La sécurité du modèle ACJT**

Ce modèle répond aux exigences de la sécurité souhaitées :

✓ **La résistance aux coalitions :** sous l'hypothèse RSA Fort, un certificat de groupe $[A_i = (a^{xi}a_0)^{1/ei} \bmod n; e_i]$ avec $x_i \in \Lambda$ et $e_i \in \Gamma$ peut être produit seulement par le manager de groupe à condition que le nombre K de certificats donnés par le manager du groupe soit polynomialement borné.

✓ **L'inforgeabilité :** seuls les membres d'un groupe sont capables de signer au ceint du groupe.

✓ **L'anonymat :** donnant une signature valide $(c, s_1, s_2, s_3, s_4, T_1, T_2, T_3)$, l'identification le signataire actuel est calculatoirement difficile pour toute autre personne que le manager du groupe.

✓ **La non-reliabilité :** Décider si deux signatures $(c, s_1, s_2, s_3, s_4, T_1, T_2, T_3)$ et $(\tilde{c}, \tilde{s}_1, \tilde{s}_2, \tilde{s}_3, \tilde{s}_4, \tilde{T}_1, \tilde{T}_2, \tilde{T}_3)$ ont été calculées par le même membre du groupe est calculatoirement difficile. Similairement à l'anonymat, le problème de lier deux signatures mène à décider si les trois logarithme discrets $\log_y T_1/\tilde{T}_1$, $\log_g T_2/\tilde{T}_2$ et $\log_g T_3/\tilde{T}_3$ sont égaux. Ceci est impossible sous l'hypothèse Diffie-Hellman Décisionnelle.

✓ **La non-diffamation :** Ni le manager du groupe ni l'un des membres ne pourra signer au nom d'un autre membre car il est impossible d'avoir n'importe quelle information sur le secret x_i à partir de a^{xi} .

✓ **La traçabilité :** le manager du groupe est capable d'ouvrir n'importe quelle signature valide et par conséquent identifier le signataire : Supposant que la signature est valide, cela implique que T_1 et T_2 sont de la forme adéquate, alors A_i peut être découvert. Et puisque le certificat du groupe [$A_i = A(x_i)$, e_i] avec $x_i \in \Lambda$ et $e_i \in \Gamma$ peut être obtenu à partir du protocole Join, A_i découvert peut être lié à une instance unique du protocole Join , donc l'utilisateur U_i qui a généré la signature va être identifié.

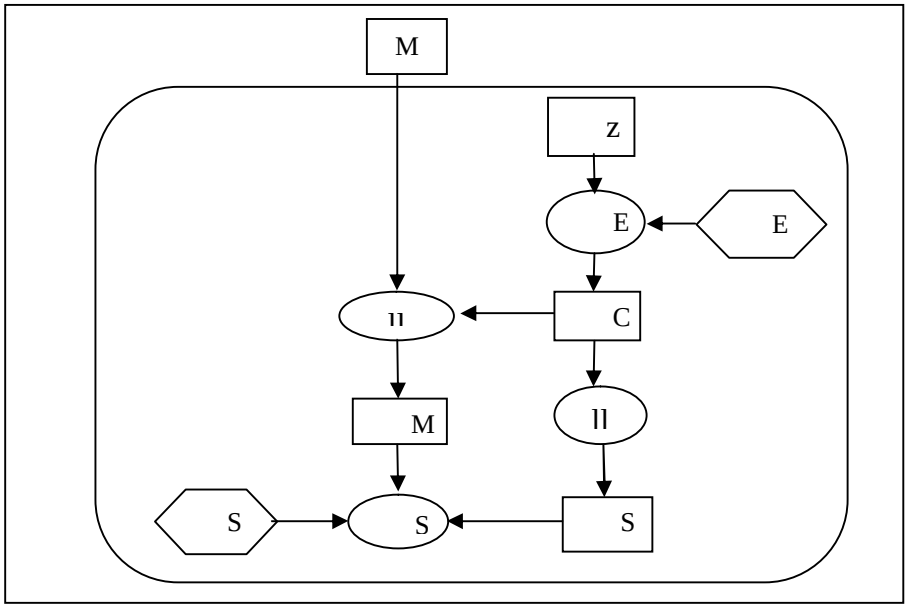
4.2.4.2. Signature de groupe avec les cartes à puce

Un nouveau schéma a été proposé par Canard et Girault [CG02] en s'appuyant sur les cartes à puce. Ce mécanisme permet d'obtenir directement l'intégrité des données (publiques et secrètes) ainsi que le programme implémenté dessus. De même que pour la confidentialité des données et des clés qui sont facilement préservées.

Pour cela, cette solution consiste en utilisation des cartes à puce et une clé privée partagée dans le groupe. D'abord, un schéma de signature ordinaire est choisi (utilisant une clé secrète sk_G et une clé publique pk_G)et systématiquement un cryptosystème sûr (avec une clé de chiffrement D_{Aut} et une clé de déchiffrement E_{Aut}) tel que le chiffré ne révèle aucune information partielle sur le texte en clair qui peut être calculé en un temps polynomial. Ensuite, le manager du groupe calcule les clés de sorte qu'il peut garder en secret D_{Aut} ou distribuer sk_G entre les membres sans les connaître, et il publie les clé publiques (pk_G et E_{Aut}).

Si un utilisateur souhaite adhérer au groupe, il doit posséder une carte à puce. Ensuite, il doit obtenir un identificateur z (qui doit être unique pour l'identifier) auprès du manager et la clé privée partagée sk_G commune à tous les membres du groupe. La carte à puce doit aussi avoir accès à tous les paramètres afin d'utiliser le cryptosystème tels que E_{Aut} et le schéma de signature déjà défini. Le manager doit garder le lien entre l'identificateur z et l'identité du membre.

Quand un utilisateur veut signer un message comme un membre du groupe, il a à utiliser sa carte à puce. Premièrement, son identificateur z est chiffré (par l'algorithme EA) avec la clé publique du manager du groupe E_{Aut} (de sorte que seul le manager aura le pouvoir de le déchiffrer). Après, Le message M est concaténé la valeur chiffrée C et le tout est signé à l'aide de l'algorithme de signature SA et la clé privée partagée sk_G . En conséquence, seuls les membres du groupe peuvent signer et n'importe qui pourra vérifier la signature avec la clé publique associée pk_G .



M : un message
ll: algorithme de concaténation
z : identité du membre
M' : concaténation de M et C
EA : algorithme de chiffrement
SA : algorithme de signature
E_{AUT} : clé de chiffrement du MG
S_G : signature du message
C : chiffré de l'identité z
SK_G : clé secrète de signature du groupe

Figure 13 : ***schéma de signature de groupe avec carte à puce***

Le vérificateur reçois la valeur chiffrée C, le message M et la signature complète, il a à vérifier que la signature provient vraiment d'un des membres du groupe.

Seul le manager du groupe peut ouvrir la signature et révéler l'identité du signataire en déchiffrant l'identificateur z avec la clé D_{Aut}.

- **La résistance aux coalitions :**

Le problème de résistance aux coalitions est facilement résolu en utilisant un dispositif résistant à la falsification. En pratique, il est impossible à deux membres de créer une nouvelle carte sans avoir accès aux données protégées. En particulier, ils n'ont aucune connaissance de la clé secrète partagée sk_G (information protégée dans leurs cartes). En plus, produire une signature sans savoir la clé privée viole l'hypothèse de sécurité du schéma précédemment défini.

❖ **La sécurité du modèle** :

Sous l'hypothèse que la carte à puce est un dispositif résistant à la falsification, le schéma proposé est considéré sûr.

- ✓ **L'inforgeabilité :** Seuls les membres d'un groupe ont la clé privée partagée du groupe sur leurs cartes à puce (grâce à leur interaction avec le manager du groupe) et par conséquent, peuvent signer au nom du groupe.

- ✓ **L'anonymat :** tous les membres ont la même clé privée de signature et leurs identificateurs sont chiffrés. alors un vérifieur ne peut pas identifier le signataire car tout membre est capable de calculer, potentiellement, la même signature et il ne pourra rien extraire de la valeur chiffrée (un cryptosystème sûr).

- ✓ **La non-reliabilité :** Les membres du groupe ont une clé partagée et un cryptosystème sémantiquement sûr. Il est donc impossible de lier deux différentes signatures.

- ✓ **La non-diffamation** : cela est due à la réalité que l'identificateur du signataire est concaténé à sa signature de groupe et que la carte à puce est un dispositif résistant à la falsification.

- ✓ **La traçabilité** : La carte chiffre toujours l'identificateur du membre du groupe, donc, le manager peut, toujours, le déchiffrer et ouvrir la signature.

4.2.4.3. Le modèle BSZ

Bellare et al [BMW03] ont proposé un modèle de signature de groupe constitué des différents protocoles standards déjà définis. Ce modèle est appliqué dans le cas des groupes statiques, où l'algorithme Join est inclus dans la phase Setup, donc, le nombre des membres du groupe est fixé et aucun nouvel utilisateur ne pourra s'adhéré.

Suite à ce modèle, Bellare, Shi et Zhang [BSZ05] ont défini un modèle de sécurité pour les groupes dynamiques en redéfinissant certaines des propriétés du BMW.

Alors, un nouveau membre peut adhérer au groupe et obtenir un certificat d'appartenance au groupe à n'importe quel moment, en exécutant un protocole Join avec le manager du groupe. De plus, le manager d'anonymat, en preuve d'honnêteté, doit fournir une preuve qu'il a correctement exécuté l'algorithme Open, cette preuve est transmise à un juge. Le juge exécute alors un nouvel algorithme Judge afin de confirmer la bonne intention du manager d'anonymat (autorité de révocation).

Le nouveau schéma de signature cette fois, comprend un nouveau participant (juge) et un algorithme de plus (Judge).

Le modèle BSZ propose une terminologie spécifique :

- Une partie de confiance pour la génération initiale des clés
- Une autorité nommée : émetteur
- Une autorité nommée : ouvreur
- Un ensemble d'utilisateurs, chacun a une identité unique $i \in N$, qui peuvent devenir membres.

Le schéma est spécifier comme un tuple GS = (GKg, UKg, Join, Iss, GSig, GVf, Open, Judge) d'algorithme en temps polynomial.

Prenant : k ∈ N dénote un paramètre de sécurité, les algorithmes du schéma sont les suivants :

- GKg : dans la phase d'initialisation (Setup), la partie de confiance exécute l'algorithme de génération de clé de groupe GKg qui utilise en entrée 1^k et retourne le triple (gpk, ik, ok). La clé d'émission ik est remise à l'émetteur (issuer), la clé d'ouverture ok à l'ouvreur (opener) et la clé publique de vérification de signature gpk est rendue publique.

- UKg : un utilisateur qui veut devenir membre du groupe doit commencer par exécuter l'algorithme de génération de clé d'utilisateur UKg sur la donnée 1^k pour obtenir une paire clé privée/clé publique personnelle (upk[i], usk[i]).

- Join et Iss : une fois l'utilisateur a sa paire de clé personnelle, il s'engage dans le protocole de jointure du groupe avec l'émetteur. Les deux algorithmes interactifs Join et Iss présentent, respectivement, les deux cotés de l'interaction. L'interaction commence par un message de l'utilisateur i (demande d'adhésion). Si l'émetteur accepte cette demande, il ajoute i comme nouvelle entrée reg[i] dans sa table d'enregistrement reg. Le contenu de cette entrée est le dernier message de Iss, et si i est accepté dans le groupe, la dernière information arrivée à Join est la clé privée de signature de i (gsk[i]).

- GSig : en possession de sa clé privée de signature gsk[i], le membre i peut appliquer l'algorithme de signature de groupe GSig à un message m, il engendre alors une signature s.

- GVf : n'importe qui, maintenant, possédant la clé publique du groupe gpk, peut vérifier cette signature s à l'aide de l'algorithme GVf qui retourne la valeur 1 si la signature est valide.

- Open : L'ouvreur utilise, si nécessaire, sa clé d'ouverture ok pour exécuter l'algorithme Open sur l'ensemble (message m, signature s du message m par gpk, table d'enregistrement reg) qui retourne une paire (i, p) où i doit être un entier. Si i est supérieur ou égal à 1, l'algorithme déclare que le membre d'identité i a vraiment signer m, sinon, il déclare qu'aucun membre du groupe n'a produit la signature s. Dans ce dernier cas p est une preuve de cette réclamation qui pourra être vérifiée par l'algorithme Judge.

- Judge : cette algorithme prend en entrée : la clé publique du groupe gpk, un entier j > 0, la clé publique upk[j] d'une entité avec l'identité j (si cette entité n'a aucune clé publique upk[j]= ε), un message m, une signature valide s de m et la preuve p. il a pour but de vérifier si p est une preuve que j a produit s. cette vérification est basée sur la clé publique de j.

❖ **La sécurité du modèle**

Les propriétés de sécurité d'un schéma de signature de groupe dynamique sont regroupées en quatre :

- ✓ **La consistance (exactitude) :** une signature s est jugée valide si elle a été émise de manière honnête par un membre honnête. Et, étant donné un couple message m/ signature s, l'algorithme de traçabilité doit retrouver l'identité du membre et la preuve émise par l'ouvreur (autorité de révocation) doit être acceptée par le juge.

- ✓ **L'anonymat :** Il est strictement impossible d'extraire des informations significatives à partir d'un couple message/signature. L'attaquant doit avoir accès à la table d'enregistrement reg pour obtenir, si possible, l'identité i du membre, alors que cette table est chiffrée avec la privée de l'émetteur.

- ✓ **La traçabilité :** toute signature valide doit être révocable. Dans ce schéma, un adversaire doit être incapable de produire une signature valide de telle sorte que l'ouvreur ne pourra pas identifier l'origine.

- ✓ **La non-diffamation :** dans ce schéma, il est impossible de produire une signature de groupe valide telle qu'une autorité honnête peut l'ouvrir et produire une preuve valide que cette signature provient vraiment d'un membre honnête du groupe.

4.3. Signature aveugle

Une variante intéressante des signatures électroniques est la signature aveugle. Ce nouveau concept a été introduit par Chaum en 1982 afin de garantir l'anonymat du dépenseur dans un système de monnaie électronique. Une telle signature exige que le signataire soit capable de signer un document sans avoir la moindre connaissance sur son contenu. En plus, si le signataire obtient la paire document/signature, il doit être incapable de déterminer quand ni pour qui il a signé le document, même si il est capable de vérifier la validité de la signature. L'exemple le plus célèbre dans ce domaine, est l'utilisation de signature sur la monnaie électronique, où le document à signer est une pièce de monnaie électronique est le signataire est une banque. L'anonymat du dépenseur est retenu dans toute transaction impliquant des pièces de monnaie électronique tant que ces dernières sont aveuglement signées.

Le concept de signature aveugle a été introduit pour la première fois par David Chaum en 1982 afin d'établir un moyen de payement intraçable, où le dépenseur reste anonyme pour le commerçant et le commerçant reste anonyme pour la banque auprès de laquelle il a retiré cet argent. Ensuite, Stadler, Piveteau et Camenisch on introduit une nouvelle variante de signature aveugle [SPC95] qui représente un schéma de signature aveugle à anonymat révocable. Pointcheval et Stern [PS96] ont, en 1996, introduit modèle formel de signature aveugle et ont prouvé la sécurité du schéma de signature d'Okamoto-schnorr.

En 1997, Juels, Luby et Ostrovsky [JLO97] ont montré la sécurité d'un schéma de signature contre une attaque par plusieurs exécutions en parallèle et de manière concurrente. Deux ans après, Ramzan [Ram99] a proposé la concaténation la notion déjà existante de signature de groupe et celle de signature aveugle. La signature aveugle de groupe permet aux membres de groupe assez large de signer un message au ceint du groupe de manière sécurisée. En 2003, Popescu [Pop03] a réalisé une extension du modèle ACJT en rajoutant la propriété d'aveuglement pour permettre aux membres du même groupe de signer des messages au milieu du groupe sans connaître son contenu. Un nouveau type de signature aveugle de groupe basé sur le couplage bilinéaire a été développé en 2006 par Dake He [He06]. Sa

sécurité est basée sur le problème Diffie-Hellman Calculatoire dans le modèle à oracle aléatoire.

En 2002, Zhang et Kim [ZK02] ont utilisé un nouveau concept dans leur schéma, l'utilisation de la cryptographie basée sur l'identité en utilisant le couplage bilinéaire. Où ils ont proposé un schéma de signature aveugle et un schéma de signature d'anneau basés tous les deux sur l'identité et le couplage bilinéaire. Kiayias et zhou [KZ06] ont prouvé la sécurité de leur schéma de signatures aveugles concurrentes sous les hypothèses LRSW et Diffie-Helleman Décisionnel.

Certaines autres variantes des schémas de signatures aveugles ont été proposées comme : la signature partiellement aveugle, proposée par Okamoto [Oka06] en 2006, dont le principe est de dévoiler une partie du document à signer. Et la signature aveugle restrictive, qui permet de restreindre le choix des messages à signer pour l'utilisateur, proposée par Chen, Zhang, Mu et Susilo dans [CZMS06] en 2006.

4.3.1. Fonctionnement et construction

Le principe de ce type de signature est de signer un message m pour un utilisateur U sans que le signataire S n'aie la moindre connaissance sur le contenu de m. D'abord, le message m doit être aveuglé (m') (chiffré) par U puis envoyer au signataire. Une fois m' reçu, le signataire S utilise un des algorithmes de signature pour le signer (le valider) et le retourne (s') à son propriétaire. Quand l'utilisateur récupère la signature s' sur le message aveuglé m', il la désaveugle (déchiffre) pour avoir la signature s voulue sur son message d'origine m.

D'après cette introduction, un schéma de signature aveugle nécessite deux participants :

- Un utilisateur U qui souhaite avoir une signature valide sur un message n de son choix,
- Un signataire S qui va signer le message pour U.

En plus des participants, on peut remarquer que le schéma comporte principalement trois étapes :

- aveuglement du message,
- signature du message aveuglé
- Désaveuglement de la signature

Mais cette procédure est attachée une paire de clés de signature, donc une nouvelle étape de génération de clés doit précéder ce protocole et vers la fin, la signature doit être validée par un algorithme de vérification, ce qui implique l'existence de cinq algorithmes au total dans un tel schéma de signature.

Donc, un schéma de signature aveugle comprend les trois étapes suivantes :

- **La mise en place des clés (Setup)** : un algorithme de génération de clés. Il prend en entrée un paramètre de sécurité k et donne les paramètres du

système et la paire de clés (clé publique/clé secrète « pks/sks ») du signataire.

- **La signature (Sign)** : un protocole entre l'utilisateur U et le signataire S. il est subdivisé en trois étapes :

- **L'aveuglement (Blind)**: un algorithme qui prend en entrée le message m, un nombre aléatoirement choisi r et retourne en sortie m', le message aveuglé de m.

- **La signature aveuglée (BSign)** : un algorithme de signature électronique qui prend en entrée le message aveuglé m', la clé privée du signataire sks et retourne une signature aveuglée s'.

- **Le désaveuglement (Unblind):** un algorithme prenant en entrée la signature aveuglée s' et le nombre r, il renvoie la signature désaveuglée s du message m.

- **La vérification (Verify)** : un algorithme de vérification de signature électronique, qui prend la signature s, le message m, et la clé publique pks du signataire S et retourne 1 si la signature est valide, 0 sinon.

4.3.2. Sécurité des signatures aveugles

Dans un schéma de signature aveugle, un utilisateur interagit avec un signataire afin de résulter une signature valide. Et chacun d'entre eux a l'opportunité de casser ce schéma :

- L'utilisateur cherche à produire des signatures valides sur des messages de son choix sans interagir avec le signataire.

- Et le signataire cherche de son coté à marquer la signature de telle sorte qu'il pourra ultérieurement reconnaitre la signature et identifier l'utilisateur.

La première propriété est dite la falsification tandis que la deuxième est l'aveuglement.

- **L'aveuglement « l'indistinguabilité »:**

L'aveuglement consiste à ce que le signataire ne doit rien apprendre sur le message qu'il va signer. Il doit être, aussi, incapable de rajouter la moindre information sur la signature qui lui permettra après de la reconnaître. L'utilisateur a aussi besoin de la garantie que s'il demande plusieurs signatures, le signataire soit incapable de relier entre elles.

Afin d'aboutir à une définition formelle de l'indistinguabilité, Juels et al [JLO97] ont mené cette expérience :

Soit A un algorithme probabiliste en un temps polynomial, $b \in \{0, 1\}$un bit aléatoirement choisi inconnu de A et k un paramètre de sécurité. L'algorithme A exécute l'expérience suivante (où A contrôle le signataire et non l'utilisateur, et essaye de prédire b):

- **Etape 1** : (pks, sks) ← Gen(1^k)

- **Etape 2** : {m_0, m_1}← A(1^k , pks, sks) (A polynomial en 1^k, produit deux documents m_0 et m_1 lexicographiquement ordonnés et qui peuvent être dépendants de pks et sks)

- **Etape 3** : on note par {m_b, m_{1-b}} les deux documents {m_0, m_1}ordonnés selon la valeur du bit b qui cachée de A. A(1^k ,pk, sk, m_0, m_1) s'engage en deux protocoles interactifs parallèles, le premier avec un utilisateur U(pks, m_b) et le deuxième avec un utilisateur U(pks, m_{1-b})

- **Etape 4** : si le premier utilisateur renvoi s(m_b) et le second renvoi s(m_{1-b}) (i.e : aucun des deux n'échouent dans l'exécution du protocole) alors a reçoit les deux signatures {s_b, s_{1-b}}

- **Etape 5** : A reçoit un bit b'

Alors, la probabilité prise sur le choix de b, des nombres aléatoires de génération de clés et des nombres aléatoires de A que b'=b est au plus (1/2)+($1/k^c$) où c est une constante.

Dans une attaque d'indistinguabilité, l'adversaire est capable de corrompre un signataire et de répondre à des demandes de signature de la part d'utilisateurs honnêtes. A un moment de son choix, il sélectionne deux messages et deux utilisateurs honnêtes et s'engage dans deux protocoles de signature sans savoir quel message est envoyé à chacun des utilisateurs. Il reçoit en réponse les deux signatures dans un ordre aléatoire. Le but de son attaque est alors de déterminer la provenance de chaque signature en renvoyant un bit b' correspondant à l'ordonnancement des messages.

- **La falsification :**

La non-falsification est la garantie que l'utilisateur ne peut pas produire une signature valide soit même. En fait, la falsification existentielle sous un message choisi adaptivement est de quelque sorte la base de la signature aveugle car l'aveuglement rend difficile la vérification des messages demandés en signature. La seule façon possible de restreindre un utilisateur malicieux est de limiter le nombre d'interactions lui est autorisées avec le signataire.

La manière la plus adéquate de spécifier une exigence de non-falsification est d'empêcher un utilisateur de faire ce qui est appelé la (l, l+1)-falsification (one-more-forgery) cela signifie qu'un utilisateur ne peut pas obtenir l+1 signature après avoir eu l interactions avec un signataire.

- **Définition de la (l, l+1)-falsification (falsification supplémentaire):**

Pour tout entier l, si après l interactions avec un signataire S, un algorithme probabilistique en un temps polynomiale A parvient à obtenir une signature valide l+1, on dit que la falsification supplémentaire est une (l, l+1)-falsification pour l'entier l.

Et comme toujours, un attaquant a différentes méthodes pour effectuer sa falsification, dont on cite deux types :

- Attaque séquentielle : où l'attaquant interagit de manière séquentielle avec le signataire et c'est dans ce cas que l'attaque peut être corrompue en limitant le nombre d'interactions possible entre utilisateur/signataire.

- Attaque parallèle : l'attaquant interagit l fois en parallèle avec le signataire. Ce type d'attaque est plus fort. En effet, l'attaquant peut initier une nouvelle interaction avec l'utilisateur avant la fin des précédentes, donc il peut atteindre le nombre qui lui permet d'effectuer son attaque sans que le signataire ne s'en aperçoive.

Juels et al [JLO97] ont définit la non-falsification comme suit :

L'algorithme adversaire A exécute l'expérience suivante (cette fois A contrôle l'utilisateur et non le signataire, il essaye cette fois d'avoir une signature de plus « one-more-signature »)

- **Etape 1** : (pks, sks) ← Gen (1^k)

- **Etape 2** : A(pks) engage en plusieurs exécutions adaptatives et parallèles de protocoles interactifs avec un nombre polynomialement borné de copies de signataire(pks, sks), où A décide lui-même l'arrêt des exécutions quand il atteindra les l interactions achevées.

- **Etape 3** : A retourne l'ensemble {(m1, s1), …, (mj,sj)} et tous les couples $1 \leq i \leq j$ sont accepté par Verify (mi, si,pks).

Alors, la probabilité que j > l est au plus $1/k^c$, où c est une constante et k un entier assez grand.

4.3.3. Domaines d'utilisation des signatures aveugles

Les signatures aveugles permettent d'obtenir des signatures sur des documents sans avoir la nécessité de dévoiler leurs contenus. Grace à cet avantage, elles trouvent leurs places dans plusieurs domaines dont deux retiennent l'attention : Le vote électronique et la monnaie électronique.

❖ **Utilisation dans les votes électroniques** :

Dans un système de vote traditionnel, l'électeur suit une certaine procédure : d'abord, il s'identifie auprès d'une autorité, pour prouver son droit du vote. Il sélectionne après son vote, cache le bulletin dans une enveloppe et le met dans l'urne.

Pareil, dans un schéma de vote électronique, un schéma de signature aveugle assure les conditions nécessaires au déroulement légal d'un vote. Pour cela, il doit y avoir deux entités dans un schéma de vote électronique : un électeur et une autorité de supervision de vote nommée : Service de Tabulation Centrale « STC ». Et pour restreindre le nombre de bulletins, on suppose que le vote est de la forme « Oui » ou « Non ». Le protocole de vote est, alors, divisé en deux phases :

- **Enregistrement :** la phase d'enregistrement se fait en trois étapes :

1. L'électeur crée deux bulletins électroniques B1 et B2 (constitués d'un numéro de série et certaine information appropriée au vote). Le Bulletin B1 contient le vote « Oui » et le bulletin B2 contient le vote « Non ».

2. L'électeur, ensuite, aveugle les deux bulletins B1 et B2 et envoi les versions aveuglées au « STC ».

3. Le « STC » vérifie d'abords que cet électeur n'a pas encore voté. Si c'est le cas, il signe, alors, les bulletins aveuglés et les remet à l'électeur.

- **Vote :** Cette phase se déroule, elle aussi, en quatre étapes :

1. L'électeur désaveugle les deux bulletins B1 et B2. Il obtient deux ensembles de votes valides signés par le « STC ».

2. il choisi celui qu'il veut, il l'associe avec la signature du « STC » et le chiffre avec la clé publique du « STC ».

3. il envoi son vote au « STC ».

4. Le « STC » déchiffre le vote, vérifie que la signature est valide, vérifie, ensuite, sa base de données pour s'assurer que le numéro de série sur le vote n'a pas été utilisé précédemment (cela empêche l'électeur d'essayer de revoter une nouvelle fois). Si la vérification est bonne, le « STC » enregistre le vote et le numéro de série associé dans sa base de données.

A la fin des élections, le « STC » publie les résultats des votes (les numéros de séries des bulletins et les votes associés).

❖ **Utilisation avec la monnaie électronique** :

Chaum, dans son schéma de signature aveugle, avait projeté la création d'une version électronique d'argent. Pour atteindre ce but, il a introduit la notion de « pièce de monnaie électronique » « electronic coins) avec les signatures aveugles.

Il signala que c'était le seul moyen d'assurer l'anonymat exigé : dans la vie réelle, une pièce de monnaie électronique ne peut pas être facilement traçable par la banque ou le magasin. Aussi, deux dépenses d'un même utilisateur ne peuvent pas être reliées ensemble. Donc, deux propriétés essentielles doivent être assurées par les pièces de monnaie électronique : la non-traçabilité et la non-reliabilité.

Comme il proposa de définir une pièce de monnaie électronique comme un numéro avec un certificat (signature) produite par une banque. Elle est retirée d'une banque, dépensée par un utilisateur et déposée par un magasin.

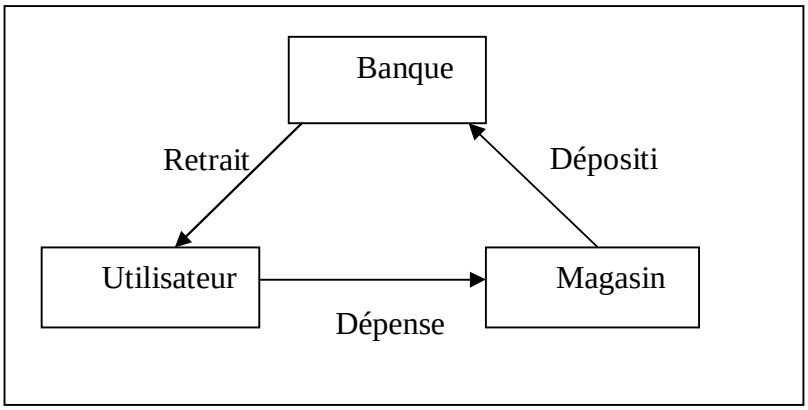


Figure 14 **:** ***le cycle de vie d'une pièce de monnaie électronique***

Protocole de base de transaction de monnaie électronique

Un protocole de manipulation de monnaie électronique (le cycle de vie d'une pièce de monnaie électronique) se déroule en trois phases :

- **Le retrait** :

    1. un utilisateur crée une pièce de monnaie électronique C. Cette pièce consiste en une chaine de bits qui spécifie certaines informations telle qu'un numéro de série et un montant.

    2. L'utilisateur et la banque, ensuite, s'engage dans un protocole de signature aveugle, qui retourne à l'utilisateur une signature valide sur une pièce de monnaie électronique.

    3. A une fin réussie du protocole, la banque retire le montant spécifié sur la pièce du compte de l'utilisateur.

- **La dépense** :

    1. L'utilisateur présente la pièce de monnaie électronique C signée par la banque pour régler un achat.

    2. Le magasin vérifie la validité de C en vérifiant la signature avec la clé publique de la banque. Si la signature est invalide, la transaction (l'achat) est annulée.

- **La déposition** :

1. A fin de récupérer son argent, le magasin présente la pièce C et la signature de la banque sur C au banquier.

2. La banque vérifie la validité de la signature sur la pièce C. si elle est valide, la banque a besoin de vérifier que la pièce C n'a pas été dépensée avant. Si la signature est valide et la pièce C n'a pas été déjà dépensée avant, la banque verse le montant spécifié sur la pièce au magasin.

3. Si les vérifications sont conformes, La transaction est finalisée par la remise des achats à l'utilisateur.

On observe à la fin de la transaction que l'anonymat de l'utilisateur est garantit devant le magasinier et le banquier.

Mais, on peut extraire deux failles potentielles sur ce schéma de monnaie électronique :

- puisque la banque signe aveuglement le document, rien n'empêche l'utilisateur de présenter à la banque un document frauduleux, comme par exemple un montant de la pièce qui dépasse largement le solde de l'utilisateur.

- et comme cette pièce de monnaie électronique est une chaine de bits, elle est facilement reproductible : ce qui permet de réutiliser la pièce une nouvelle fois.

Si une transaction se termine, il est strictement impossible de poursuivre l'utilisateur tant que son identité est complètement anonyme. Heureusement, certaines techniques existent pour résoudre ces failles.

Pour résoudre le problème de document frauduleux, la banque dispose de deux techniques possibles :

1. La banque peut utiliser différentes clés publiques associées aux différents montants d'argent. Donc pour signer une pièce de monnaie électronique avec un montant bien précis, la banque doit utiliser la clé publique associée à ce montant. Alors l'utilisateur ne pourra faire estimer la pièce plus que ce qu'elle vaut réellement.

2. La banque et l'utilisateur peuvent exécuter un protocole de « coupe et choisi ». où l'utilisateur prépare un nombre de pièces de monnaie électronique identique à faire signer auprès de la banque (soit disant 10 : C1,C2,…,C10) la seule différence entre eux est le numéro de série. Ensuite, il les aveugle et les envoie à la banque. La banque prend toutes les pièces sauf une et demande à l'utilisateur le facteur d'aveuglement de ces pièces. Elle désaveugle les pièces et les vérifie. Si elles sont toutes de la forme appropriée, la banque signe la pièce restante et l'envoie à l'utilisateur.

   La banque aura une haute assurance que la pièce restante est dans le format correct, car si l'utilisateur y introduit au milieu une pièce frauduleuse, il sera capturé avec une probabilité de 9/10.

Il reste la deuxième faille : la double dépense de la pièce. Après tout, la pièce C n'est qu'une chaine de bits qui peut être dupliquée. Donc il faut avoir un moyen de détecter et interdire la double dépense d'une pièce C.

La solution possible est la possession d'une sorte de liste des pièces de monnaie électronique déjà dépensées au niveau de la banque. Après, et pondant chaque transaction, à la réception d'une pièce C d'un vendeur, la banque doit vérifier si cette pièce a été déjà dépensée. Si c'est le cas, la banque informe le vendeur, et le vendeur refuse de donner la marchandise à l'acheteur. Ce schéma est sûr mais couteux car le vendeur doit être relié à la banque en permanence pour vérifier la pièce C, et si l'utilisateur double dépense la pièce il

doit être capturé immédiatement, sinon il ne le sera jamais puisque son identité est complètement anonyme.

4.3.4. Quelques schémas de signatures aveugles

Parmi les schémas de signature aveugle proposés à travers le temps, on a choisi de présenter certains d'entre eux.

- **Le schéma de signature aveugle RSA**

Le premier schéma de signature aveugle proposé par Chaum à été fondé sur la signature traditionnelle RSA qui donne une signature s en calculant l'exponentiation du haché h d'un message m avec l'exposant secret d , le tout modulo un module publique n

$$s(h(m)) \equiv (h(m))^d \pmod n$$

La validité de la signature s est vérifiée en calculant l'exponentiation de la signature s avec l'exposant publique e, le tout modulo le module n et comparer le résultat au haché original du message m :

$$(s(h(m)))^e \equiv h(m) \pmod n$$

La version aveugle de RSA comporte les trois phases suivantes :

- Setup : la génération des clés de signature :
  - Choisir aléatoirement deux entiers premiers p et q
  - Calculer n = pq , le module de RSA
  - Choisir e : un entier premier avec $\varphi(n) = (p-1)(q-1)$
  - Calculer $d = e^{-1} \bmod \varphi(n)$

  La paire de clés de signature (sks, pks) est (d, (n,e)).
- Sign : la phase de signature se compose de trois étapes :
  - Blind : l'aveuglement du message :
    - ✓ L'utilisateur U choisi aléatoirement un nombre r tel que r soit premier avec n.
    - ✓ Il calcule $h' = h(m)r^e \bmod n$
    - ✓ Il envoie h' au signataire S
  - BSign : Le signataire S signe alors le message aveuglé :
    - ✓ Il calcule $s' = h'^d \bmod n = (h(m)r^e)^d \bmod n = h(m)^d r \bmod n$
    - ✓ Il envoie s' à U

– Unblind : le désaveuglement de s' :

U reçois la signature s', il calcule ensuite $s = s'r^{-1} \bmod n = h(m)^d$. il envoie après le couple (s,m) à son destinataire.

- Verify : La vérification :

Le destinataire vérifie que $s^e = h(m) \bmod n$.

➢ **Le schéma de signature aveugle à anonymat révocable**

La signature aveugle à anonymat révocable permet soit de relier une signature à une session où elle a été réalisé soit identifier une signature en reconnaissant sa session. Un tel schéma est spécifié par le tuple (G_s, G_{RA}, G_U , Sig , Vf , R_{Sig} , R_{id} , M_{sig} , M_{id}):

- G_S est un algorithme de génération de clés de signature, il prend en entrée le paramètre de sécurité k et retourne la paire clé privée/clé publique du signataire (sks/pks) ← G_S (1^k).

- G_{RA} est l'algorithme de génération de clé de révocation. Il prend en entrée la clé publique pks du signataire et le paramètre de sécurité k et retourne une paire clé privée/clé publique de révocation (rsk/rpk) ← G_{RA}(1^k , pks) .

- G_U est l'algorithme de génération de clé de l'utilisateur. Il prend en entrée la clé publique pks, le paramètre de sécurité k et l'identité I_{id} de l'utilisateur U et retourne la paire clé publique/clé privé de l'utilisateur U (usk_{id}, upk_{id}) ← G_U(1^k , pks).

- Sig : un protocole interactif entre le signataire S qui prend sa clé privé sks et l'utilisateur U qui prend son message m, sa clé privée et son identité I_{id}. Si S accepte le protocole, il retourne une transcription du protocole $view_i$. si U accepte le protocole, il retourne le couple (m,s), où s est la signature sur m, sinon le protocole échoue.

- Vf : un algorithme de vérification. Il prend en entrée un message m, une signature s et la clé publique du signataire pks et il retourne 1 si s est valide, et 0 sinon.

- R_{Sig} : un algorithme de révocation. Il prend en entrée la transcription $view_i$ du signataire durant la session cible et la clé de révocation rsk, et retourne l'identificateur de signature I_{sig} qui identifie la signature obtenue durant la session cible en plus d'une preuve p1 que l'identificateur a été correctement calculé (I_{sig} , p1) ← R_{Sig} ($view_i$, rsk)

- R_{id} : un algorithme de révocation qui prend en entrée une paire message cible/signature (m/s)et la clé de révocation rsk et retourne l'identificateur I_{id} de la session à partir de laquelle la paire message cible/signature a été obtenue en plus d'une preuve p2 que I_{id} a été correctement calculé (I_{id} , p2) ← R_{id}((m/s),rsk).

- M_{sig} (resp. M_{id}) un algorithme de liaison qui examine si I_{sig} (resp. I_{id}) correspond à une paire message/signature (resp. une identité I_{id} d'un utilisateur U) ou non et si la preuve est valide. Il retourne 1 s'ils correspondent et la preuve est valide , et 0 sinon).

➢ **Schéma de signature partiellement aveugle**

Dans un scénario résultant une signature partiellement aveugle, le signataire et l'utilisateur sont supposés ayant agréé sur une pièce d'information commune notée *info.* Dans certaines applications, *info* peut être décidée par le signataire ou simplement envoyée par l'utilisateur. Autrement, *info* est externe du schéma de signature.

Un schéma de signature partiellement aveugle est donné par quatre algorithmes (G, S, U, V) :

- G : un algorithme probabilistique en un temps polynomial. Il prend en entrée un paramètre de sécurité k et retourne une paire clé privée/clé publique (pk/sk)

- S et U : une paire d'algorithmes interactifs. U prend en entrée info et la clé publique pk et le message m et S prend info et la clé privée sk. S et U s'engagent dans le protocole de signature et s'arrêtent en un temps polynomial en k (pour k suffisamment large). quand ils s'arrêtent, S retourne *completed* (pour le protocole) ou *non-completed* , si S retourne *completed* U retourne une paire (m, s) .

- V : un algorithme en un temps polynomial, prend en entrée (pk, *info*, m, s) et retourne soit *accept* ou *reject*.

Chapitre 5 :

Signatures d'Anneau et proposition d'un nouveau schéma

5.1. Signature d'anneau

Dans la notion de signature de groupe, présentée auparavant, un manager de groupe crée son groupe d'utilisateurs et leur distribue des clés de signature spécialement conçus. Les membres du groupe peuvent utiliser ces clés pour signer des documents anonymement au nom du groupe. Les signatures produites sont indistinguables aux vérificateurs mais claires pour le manager du groupe qui peut ouvrir les signatures et dévoiler les identités des signataires malintentionnés. En fait, les signatures de groupe sont un outil approprié dans le cas où les membres sont d'accord pour coopérer (pour s'enregistrer, être inscrits dans la liste, etc.)

Un nouveau concept, quelque peu différent des signatures de groupe est la signature d'anneau (signature de cercle). Une signature d'anneau est un procédé qui n'exige ni de manager de groupe, ni phase d'initialisation ni enregistrement de membres. Un tel schéma spécifie un ensemble de signataires potentiels et procure une preuve que le vrai signataire appartient réellement à cet ensemble sans dévoiler son identité. Ce schéma, alors, assure l'anonymat du signataire, où le vérificateur ne pourra rien apprendre sur l'identité du signataire et sans qu'il n'y ait de juge ou une certaine autorité pour lever cet anonymat. L'anonymat, cette fois, est alors inconditionnel.

Les signatures de groupes représentaient un outil de signature anonyme formellement fiable et sûr dans le cas où les membres sont parfaitement d'accord d'appartenir au groupe mais dans les signatures d'anneau, les membres n'ont même pas la moindre idée sur l'application où leurs clés publiques vont être impliquées. Pour cela, les différents schémas de signature d'anneau avaient pour but principale de délimiter la puissance du signataire.

Ces schémas assez nombreux, commençaient par celui de Rivest, Shamir et Tauman en 2001 [RST01] qui ont proposé la première idée de donner la possibilité à chaque signataire de créer son propre anneau de signature sans avoir besoin de l'agrément des membres, de signer anonymement sans qu'il n'y ait la possibilité de lever cet anonymat.

Bresson, Stern et szydlo, en 2002, [BSS02] ont proposé un nouveau schéma de signature d'anneau connu sous le nom de signature d'anneau à seuil qu'ils l'ont appliqué à un groupe AD-Hoc. Leur signature prend en entrée un certain seuil t et retourne dans chaque signature une preuve qu'au moins t membres de l'anneau peuvent authentifier le message. Cela empêche les délations isolées.

En 2003, Lv et Wang [LW03] ont développé un nouveau schéma permettant au signataire de prouver au vérificateur que c'est lui le propriétaire d'une signature et le vérificateur pourra déterminer la validité de cette preuve, c'est le schéma de signature d'anneau vérifiable. Une année après, Liu, Wei et Wong [LWW04] ont présenté le schéma de signature d'anneau reliable. Cette nouvelle propriété permet au vérificateur de déterminer si deux signatures ont été produites par la même personne. Ce schéma a été amélioré après par Tsang, Wei, Chan, et al [TWC$^+$04] en un schéma de signature d'anneau reliable à seuil.

Tsang et Wei [TW05] ont réussit en 2005 à faire réduire la taille de la signature d'anneau qui dépendait de la taille de l'anneau en une valeur constante pour tous les anneaux. Ensuite, ils l'ont étendu sur la propriété de reliabilité.

5.2. Construction d'un schéma de signature d'anneau

Comme déjà mentionné, dans une signature d'anneau, il n'existe pas de phase de création de groupe, et quand un utilisateur veut signer un message, il n'a pas besoin de s'identifier auprès d'un manager, mais il crée lui-même son anneau. Il choisit librement les membres qui l'intéressent pour sa signature, et à chaque fois il aura besoin d'une signature différente, il crée un autre anneau de signature. Les membres choisis pour un certain anneau doivent avoir chacun une paire de clés de signature (pk/sk) dont il publie la clé publique.

Un anneau R est défini par R = (pk1, pk2, …, pkn) où R[i]=pki (ensemble de clés publiques des membres de l'anneau R) et R[s] donne la clé publique de notre signataire. R doit être composé d'au moins deux clés publiques distinctes. Toutes les clés publiques de l'anneau doit être différentes.

Différemment d'un schéma de signature de groupe, un schéma de signature d'anneau ne comporte que deux algorithmes probabilistes : Ring-Sign et Ring-Verify.

- **Ring-Sign(m , R, sks)** : un algorithme probabiliste qui prend en entrée le message m à signer, l'anneau R et la clé secrète du signataire sks, et retourne une signature d'anneau s. La signature s doit contenir l'anneau R.

- **Ring-vérify(m, s) :** un algorithme déterministe qui vérifie la validité de la signature s sur le message m, il retourne 1 si la signature est valide et 0 sinon.

5.3. Sécurité des signatures d'anneau

L'absence du manager et de l'autorité de révocation simplifie le modèle de sécurité des signatures d'anneau. On distingue alors deux propriétés nécessaires : la non-falsification et l'anonymat.

5.3.1. La non-falsification :

Il est calculatoirement impossible pour un adversaire de produire une signature d'anneau au ceint d'un anneau dont il ne fait pas partie. La définition d'un schéma de signature d'anneau exige l'existence de la clé publique associée à la clé privée de signature parmi les clés publiques qui composent notre anneau R.

5.3.2. L'anonymat **:**

Il est impossible pour un vérificateur de déterminer l'auteur d'une signature donnée. La probabilité de découvrir la clé publique qui vérifie la signature est très négligeable. La définition d'anonymat pour les signatures d'anneau considère le cas où seuls deux membres de l'anneau concerné ne sont pas corrompus. L'adversaire doit alors décider qui de ces deux membres est à l'origine de la signature.

5.4. Proposition d'une Signature d'anneau à anonymat révocable

On a souligné les différences entre les deux notions : les signatures d'anneau permettent de se passer complètement de l'infrastructure du groupe et ne demandent ni manager, ni enregistrement ou autres …etc. En revanche, l'absence d'autorité de

révocation enlève toute barrière et ne peut prévenir contre une utilisation abusive, comme par exemple la divulgation anonyme d'informations sensibles, sujettes à scandale.

5.4.1. Description du schéma de Rivest et al.

Le véritable article fondateur de la notion de signature d'anneau est celui de Rivest et al. [RST01], qui est basé sur la notion des fonctions combinantes.

Fonction combinante : une fonction combinante $C_{k,v}$ (y1,…,yn) prend en entrée une clé k, une valeur d'initialisation v et une liste de valeurs arbitraires de même longueur ℓ. Elle renvoie une unique valeur $z \in \{0, 1\}^{\ell}$, telle que pour tout k, v, tout indice s et toutes valeurs fixées de $\{yi\}i\neq s$, $C_{k,v}$, vue comme une fonction de y_s, soit une permutation sur $\{0, 1\}^{\ell}$. De plus cette permutation est efficacement calculable, ainsi que son inverse.

Rivest et al. [RST01] proposent une fonction combinante basée sur un schéma de chiffrement symétrique E modélisé par une famille de permutations aléatoires. Par conséquent, un tel schéma s'analyse dans le modèle du chiffrement idéal. On définit donc :

$$z = C_{k,v}(y1, \ldots, yn) = E_k(yn \oplus E_k\,(yn{-}1 \oplus E_k(\ldots \oplus E_k(y1 \oplus v) \ldots))).$$

Pour tout indice s, on peut facilement vérifier que $C_{k,v}$ est une fonction combinante en réécrivant l'équation précédant de la manière suivante :

$$ys = E_k^{-1}(y_{s+1} \oplus \ldots E_k^{-1}(yn \oplus E_k^{-1}(z)) \ldots) \oplus E_k(y_{s-1} \oplus \ldots E_k(y1 \oplus v) \ldots))).$$

- **Schéma donné par Rivest et al[RST01].**

On désigne par ℓ, ℓ_b, ℓ_0 trois paramètres de sécurité. On considère un schéma de chiffrement symétrique parfait E défini sur $\{0, 1\}^{\ell}$ et utilisant des clés de ℓ_0 bits, ainsi qu'une fonction de hachage H, qui envoie des chaînes arbitraires sur des chaînes de ℓ_0 bits, et que l'on modélise par un oracle aléatoire. En fait, on se sert de H pour définir la clé symétrique du schéma de chiffrement E. Enfin, on suppose que chaque joueur P_i utilise un schéma de signature sûr basé sur une permutation à sens unique et à trappe, telle que RSA; dans ce cas, on suppose que le module a une taille $\ell_b < \ell$; typiquement, on prendra $\ell - \ell_b \geq 160$.

Le schéma proposé par Rivest, Shamir et Tauman [RST01] se base sur la fonction combinante de la façon suivante. Dans ce schéma, les entrées y_i de la fonction sont calculées par $f_i(x_i)$ pour des x_i choisis aléatoirement dans $\{0, 1\}^{\ell b}$. Une signature d'anneau sur un message m consiste en un (n + 1)-uplet (v, x1, . . . , xn). En posant :

$$z = C_{H(m),v}(f_1(x_1), \ldots, fn(xn)),$$

la signature est valide si et seulement si z = v.

Pour tout message m, tout ensemble de valeurs v et $\{xi\}i\neq s$, on peut calculer efficacement la valeur y_s telle que la fonction combinante renvoie v. Utilisant alors la clé secrète comme trappe pour la fonction f_s, le joueur P_s (le signataire réel) est capable de calculer x_s tel que :

$$fs(xs) = y_s,$$

complétant ainsi la signature.

Cependant, on remarque que les modules RSA impliqués dans ce processus sont différents. Une adaptation doit donc être faite pour pouvoir les combiner efficacement avec la fonction C. Pour cela, on étend la permutation à trappe RSA $fi(x) = x^{ei} \bmod ni$ à un domaine plus vaste, commun pour tous les indices. Plus précisément, on définit :

$$\text{Pour } x \in \{0, 1\}^{\ell}, \qquad gi(x) = \begin{cases} q_i n_i + f_i(r_i) & \text{si } (q_i + 1)n_i \leq 2^{\ell}, \\ x & \text{sinon ,} \end{cases}$$

Où : $x = q_i n_i + r_i$, avec $0 \leq ri < ni$.

Autrement dit, l'entrée x est découpée en tranches de ℓ_b bits qui passent alors dans la fonction RSA f_i. La probabilité qu'une entrée aléatoire soit inchangée devient négligeable lorsque $\ell - \ell_b$ croît.

En effet, la condition pour que :

$$g_i(x) = x$$

est que

$$q_i n_i + n_i \geq 2^{\ell} \qquad \text{soit} \qquad x \geq 2^{\ell} - n + r > 2^{\ell} - 2^{-\ell b}.$$

La proportion d'entrées x inchangées est donc au plus $2^{\ell b - \ell}$.

Si les permutations à sens unique et à trappe sont des fonctions RSA d'exposant 3, le schéma est particulièrement efficace : il ne demande qu'une exponentiation modulaire (et un nombre linéaire de multiplications) pour signer, et deux multiplications par membre pour vérifier.

- **Preuve de sécurité**

On peut prouver l'anonymat inconditionnel du schéma ci-dessus, au sens de la théorie de l'information, c'est-à-dire en montrant que même un adversaire doté d'une puissance de calcul infinie ne peut deviner l'identité du signataire avec une probabilité meilleure que 1/n (n étant le nombre de joueurs dans l'anneau). Ceci est due au fait que, pour tout k = H(m) et z=v, l'équation précédente a exactement $(2^{\ell})^{n-1}$ solutions, chacune pouvant être choisie avec la même probabilité, indépendamment de l'identité du signataire.

L'infalsifiabilité de la signature d'anneau est basée sur la difficulté du problème suivant, en l'occurrence l'inversion d'une permutation RSA étendue gi, définie au paragraphe suivant.

- **RSA étendu**

Etant donné un module ni de l^b bits, un entier y de ℓ bits et un exposant e premier avec $\varphi(ni)$, déterminer un entier x de ℓ bits tel que :

$$gi(x) = y,$$

avec gi définie auparavant.

La difficulté de ce problème algorithmique est clairement équivalente à l'inversion de la permutation RSA sous-jacente, puisque seule la connaissance de la trappe pour fi permet d'inverser gi sur une fraction non négligeable des entrées.

La preuve de sécurité se place, comme on l'a dit, dans le modèle du chiffrement idéal. On considère donc une famille $(E_k)_k$ de permutations aléatoires sur ℓ bits, utilisant des clés de l_0 bits, et dont l'accès est modélisé par un oracle de chiffrement (interrogeable aussi bien pour E_k que pour E_k^{-1}). L'idée générale de la preuve est de réduire un attaquant A contre le schéma de signature (un falsificateur) en un attaquant contre la permutation RSA étendue. Pour cela, on considère un simulateur recevant en entrée un module n, un exposant e, une valeur $y \in \{0, 1\}^{\ell}$ et cherchant à extraire la racine e-ième de y. Pour y parvenir, le simulateur va tenter d'insérer y comme « saut » (gap) entre deux fonctions E consécutives le long de l'anneau. Ainsi, le « OU exclusif » (XOR) entre l'entrée et la sortie (respectivement) de ces deux fonctions sera égal à y et, avec probabilité non négligeable, l'attaquant aura à extraire la racine de y pour exhiber sa falsification.

Une telle insertion est faisable à l'indice i si la requête pour une des permutations arrivant en i est faite après l'autre requête arrivant ou partant du même indice (rappelons que les requêtes peuvent être faites dans les deux sens E ou E^{-1}).

Les algorithmes de signature et de vérification donnés par le schéma de Rivest et al.[RST01] sont les suivants :

- **Génération de signature d'anneau**

Etant donné un message m, sa clé secrète k et la séquence de clés publiques des membres de l'anneau : p1,p2,…, pr, le signataire calcule sa signature comme suit :

1. choisi une clé : le signataire, d'abord, calcule une clé symétrique k en hachant le message m :

$$k = h(m)$$

2. ensuite, il choisi aléatoirement une valeur $v \in \{0,1\}^b$,

3. il choisi aléatoirement xi pour tout autre membre de l'anneau $1 \leq i \leq r$, $i \neq s$, $xi \in \{0,1\}^b$ et calcule

$$yi = gi(xi).$$

4. il résout ensuite l'équation d'anneau pour obtenir ys :

$$C_{k,v}(y1, \ldots, yn) = v$$

Par supposition, en donnant des valeurs arbitrairement aux yi, il y a une valeur unique pour ys qui résous l'équation.

5. il utilise sa valeur de trappe pour inverser gs sur ys pour obtenir xs :

$$xs = g_s^{-1}\ (ys).$$

6. il retourne sa signature sur le message m :

$$(p1,p2,\ldots,pr\ ;v\ ;x1,x2,\ldots,xr).$$

- **Vérification de la signature d'anneau**

Un vérificateur peut vérifier la signature reçue :

$$(p1,p2,\ldots,pr\ ;v\ ;x1,x2,\ldots,xr).$$

Sur le message m comme suit :

1. Pour tout i = 1,2, …,r le vérificateur calcule

$$yi = gi\ (xi)$$

2. Il hache le message m pour obtenir k

$$k = h(m)$$

3. il vérifie l'équation d'anneau en utilisant les yi :

$$C_{k,v}(y1, \ldots, yn) = v$$

Si cette équation est satisfaite, le vérificateur accepte la signature comme valide, sinon il la rejette.

5.4.2. Un nouveau schéma avec révocation d'anonymat

L'anonymat des signatures d'anneau est dit parfait, et comme l'absence d'un manager élimine la possibilité de révocation de cet anonymat, on a prévu et concrétisé une idée de développement du schéma de signature proposé par Rivest et al. [RST01] afin d'aboutir à un anonymat révocable dans les signatures d'anneau.

- **Schéma proposé**

Basé sur le schéma proposé par Rivest et al. [RST01], la construction de l'anneau doit être spontanée, telle que le signataire spécifie les membres de son anneau sans interaction avec les différents membres qu'il a choisi et même sans avoir leurs autorisations antérieures. Une condition sur le choix des membres non signataires de l'anneau est leur possession de deux paires différentes de clés (clé privé / clé publique), où l'utilisation des deux clés publiques de chaque membre est strictement indispensable dans les différents processus de notre schéma, soit $p \in P$ et $r \in R$.

Afin d'assurer la propriété de révocation, à la construction de l'anneau, le signataire doit désigner un ensemble d'autorités de révocation, de sorte que chaque membres de cet

ensemble (autorité de révocation) soit capable d'identifier le signataire actuel d'une signature valide déjà vérifier.

Notre schéma, est ainsi donné par les algorithmes suivants :

- **Sign(P, Sk, R, m) :** un algorithme de signature d'anneau qui prend en entrée le message m, la clé secrète du signataire Sk et les deux ensembles de clés publiques des membres de l'anneau P et R, et retourne une signature Sig.

- **Verify(P, m, R, Sig) :** est l'algorithme de vérification de signature. Il prend en entrée le message m, la signature Sig et les deux ensembles de clés publiques P et R. il retourne la valeur 1 si la signature est valide et 0 sinon.

- **Revoke(P, Sk', R, Sig) :** est un algorithme d'ouverture de la signature. Il prend en entrée les deux ensembles de clés publiques P et R, la signature Sig et la clé secrète de l'une des autorités de révocation Sk'.(Sk' est la clé secrète correspondante la clé publique du membre autorité de révocation utilisée dans la signature) et il retourne la clé publique du signataire de l'ensemble P (celle qui désigne le signataire).

➢ **Construction du schéma**

Tout au long de la description du schéma, nous prenons les hypothèses suivantes :

- la taille de l'anneau est t individus et

- le signataire choisi est s où $1 \leq s \leq t$.

Les différents algorithmes utilisent une fonction de permutation à trappe RSA, une fonction de chiffrement symétrique, une fonction de hachage et une fonction combinante.

➢ **Fonction de permutation à trappe RSA**

Comme il a été déjà signalé, tout membre de l'anneau doit avoir une clé publique RSA pi=(ni,ei) qui spécifie la fonction de permutation à sens unique et à trappe f_i de $\mathbb{Z}_{ni}$:

$$f_i(x) = x^{ei} \pmod{n_i}$$

On suppose que seul le signataire i sait comment inverser la permutation efficacement, en utilisant l'information de trappe dont il est le seul à connaitre la clé secrète.

Sachant que les clés utilisées ne parviennent pas du même générateur, ce qui implique l'existence de domaines de différentes tailles (le module ni de génération de chaque paire de clés diffère d'un membre à un autre). Le problème s'impose au niveau de combinaison des différentes signatures. La solution réside dans l'extension des permutations à trappe afin d'avoir le même domaine $\{0,1\}^b$, où 2^b est une puissance de 2 plus large que tous les modules n_i. Notre fonction de permutation à trappe RSA est appliquée sur le deuxième ensemble de clés publiques utilisées R.

Pour toute permutation à trappe f sur $\mathbb{Z}_n$, on définie une permutation à trappe g sur $\{0,1\}^b$: pour chaque b-bits de m on définie deux entiers positifs q et r tel que Ri = qini + ri et $0 \leq ri \leq n$ puis :

$$Yi = g(Ri) = \begin{cases} qini + f(ri) & si\ (qi + 1)ni \leq 2^b, \\ Ri & sinon\ , \end{cases}$$

- **Fonction de chiffrement symétrique**

On utilise un algorithme de chiffrement symétrique publiquement définie E tel que pour chaque clé k de longueur ℓ , la fonction E_k est une permutation sur une chaine de b-bits de longueur. Ici on utilise le modèle de chiffrement idéal où toutes les parties ont accès à un oracle procurant des réponses aléatoires aux différentes requêtes de la forme $E_k(x)$ et $E^{-1}{}_k(y)$ avec la condition que E_k soit une permutation et que les réponses soient consistantes avec leurs précédentes.

- **Fonction de hachage**

L'utilisation de la fonction E, nécessite une clé de permutation k qui est obtenue par une fonction de hachage exécutée sur le message m. cette fonction résistante aux collisions retourne un haché de longueur fixe ℓ.

- **Fonction combinante**

On utilise une famille de fonctions combinantes $C_{k,v}$ (y1, y2, ….,yt) sur une valeur d'initialisation *v* choisie arbitrairement avec la valeur k du haché de m et les valeurs yi. Chaque fonction combinante utilise E_k comme sous procédure et produit en sortie une valeur z.

- **Le schéma de signature**

Apres avoir défini les sous procédures de génération et de vérification de signature d'anneau, il faut définir les procédures globales.

- **Génération de signature (Sign)**

Etant donné un message m, sa clé secrète k et la séquence de clés publiques des membres de l'anneau : p1,p2,…, pt, le signataire calcule sa signature comme suit :

1. choisi une clé : le signataire, d'abord, calcule une clé symétrique k en hachant le message m :

$$k = h(m)$$

2. ensuite, il choisi aléatoirement une valeur $v \in \{0,1\}^b$,

3. il utilise Ri pour tout autre membre de l'anneau $1 \leq i \leq r$, $i \neq s$, $Ri \in \{0,1\}^b$ et calcule

$$yi = gi(Ri).$$

4. il résous, ensuite l'équation d'anneau pour obtenir ys :

$$C_{k,v}(y1, \ldots, yt) = v$$

Par supposition, en donnant des valeurs arbitrairement aux y_i, il y a une valeur unique pour y_s qui résous l'équation.

5. il utilise sa valeur de trappe pour inverser gs sur ys pour obtenir Rs :

$$R_s = g_s^{-1}\ (y_s).$$

6. il retourne sa signature sur le message m :

$$(p1,p2,\ldots,pt\ ;v\ ;R1,R2,\ldots,Rt).$$

- **Vérification de signature (Verify)**

Un vérificateur peut vérifier la signature reçue :

$$(p1,p2,\ldots,pt\ ;v\ ;R1,R2,\ldots,Rt).$$

Sur le message m comme suit :

1. Pour tout i = 1,2, …,t le vérificateur calcule

$$yi = gi\ (Ri)$$

2. Il hache le message m pour obtenir k

$$k = h(m)$$

3. il vérifie l'équation d'anneau en utilisant les yi :

$$C_{k,v}(y1, \ldots, yt) = v$$

Si cette équation est satisfaite, le vérificateur accepte la signature comme valide, sinon il la rejette.

- **Ouverture de signature (Revoke)**

La signature pourra être ouverte afin de découvrir l'identité du signataire en cas de conflit.

L'autorité de révocation a la possibilité de reproduire la même signature en possédant les trappes de permutation exactes.

Les différentes fonctions utilisées dans le processus de signature sont inversibles. Une autorité de révocation prend en entrée les deux ensembles de clés publiques P et R, la signature Sig et sa clé secrète Sk' correspondante à la clé publique utilisée dans la signature (Sk' ϵ R). On procédant par élimination, l'autorité de révocation calcule les yi correspondants aux paires de clés reçues avec la signature et vérifie laquelle ne satisfait pas l'équation d'anneau :

$$C_{k,v}(y1, \ldots, yt) = v$$

et il retourne la clé publique (y_s) $y_s \in P$ qui ne correspond à aucun membre, et le membre manquant de l'ensemble de vérification est notre signataire actuel.

CONCLUSION

On relève, des situations concrètes et réalistes tels que les applications de e-government, de e-commerce ou toute application financière nécessitent des exigences particulières de sécurité : l'anonymat. Dans ce contexte, ce travail nous a permis d'étudier les différentes techniques de sécurisation des transmissions des informations pour enfin proposer un nouveau schéma de signature d'anneau.

A cet effet, nous avons pris en considération, toutes les notions et primitives de cryptographie, à savoir :

- Les techniques de chiffrement

- Les fonctions de hachage

- les signatures numériques

Nous avons également étudié, les techniques de signatures anonymes qui sont : les signatures de groupe découvertes par Chaum et Van Heyst en 1991, développés ensuite dans de nombreux laboratoires de recherche. Depuis les signatures aveugles, n'ont cédé la place à aucune autre technique grâce à leur utilisation bien particulière : un signataire est sensé signer des documents sans savoir la moindre information sur leurs contenus. Il sait seulement que ces documents obéissent au règlement interne régissant ces propriétés.

Les signatures d'anneau représentent le dernier concept étudié. Ce sont des variantes de signatures de groupe avec quelques petites nuances, dont l'absence d'un manager de groupe et l'anonymat du signataire.

Cette étude nous a révélé qu'il existe une trappe le protocoles de signature de Rivest et al.[RST01]. A partir de cette analyse, nous avons développé un nouveau modèle de signature d'anneau à anonymat révocable qui s'appuie sur les trois algorithmes :

- la signature (Sign) où on a modifié l'algorithme proposé par Rivest et al.[RST01] en utilisant une donnée identifiant un utilisateur du système au lieu d'une donnée aléatoire.

- La vérification (Verify) de la signature reçue avec les données jointes

- Ouverture (Revoke) de la signature valide déjà vérifiée afin de dévoiler l'identité du signataire.

Perspectives futures :

Pour tirer des conclusions fiables sur la technique d'anonymat développé, ce travail nécessite :

- Simulation du modèle proposé.

- Implémentation du modèle sur une infrastructure réelle d'anonymat.

RESUME

La question d'anonymat se pose dans de nombreuses situations et tout particulièrement dans les transactions électroniques sur des réseaux de grande envergure tel que Internet. Ce concept peut être défini comme l'état d'un être non identifiable au milieu d'un ensemble d'individus appelé : ensemble d'anonymat. Autrement, c'est la propriété de cacher l'identité, procurant une totale élimination des informations pouvant identifier un individu. De nombreuses solutions cryptographiques ont été apportées afin de renforcer la confiance des utilisateurs. C'est dans ce contexte que s'inscrivent les principaux objectifs de ce travail qui consistent à élaborer une nouvelle technique dans le domaine des signatures anonymes.

Dans ce mémoire, nous présentons d'abord les différentes techniques cryptographiques qui forment la base pour les signatures anonymes. Ces dernières subdivisées en signatures de groupe, signatures aveugles et signatures d'anneau. Ce travail se focalise sur la création d'un nouveau schéma de signature d'anneau qui procure une nouvelle procédure permettant l'ouverture de la signature et la révocation de l'anonymat du signataire en cas de besoin.

Mots clés : cryptographie, signature anonyme, signature d'anneau, révocation d'anonymat.

ABSTRACT

The problem of anonymity is asked in many situations, particularly in the field of electronic transactions over the internet. This concept can be defined as the state of being unidentified on behalf of a set of individuals called the anonymity set. Otherwise, it is the property of hiding the person's identity offering a total elimination of personal information that can identify a person. Many cryptographical solutions are proposed to reinforce the users reliance. In this context rely our principal objectives that consist on elaborating a new technique in the domain of anonymous signatures.

In this paper, we firstly present the different cryptographic techniques that present the base of the anonymous signatures. This last concept is divided on three subgroups: group signatures, blind signatures and ring signatures. Our work is on the creation of new ring signature scheme that provides a new process helping the verifiers for opening signatures and revoking the signer anonymity if necessary.

Key words: Cryptography, anonymous signature, ring signature, anonymity revocation.

BIBLIOGRAPHIE

[ACJT00] G. Ateniese, J. Camenisch, M. Joye, G. Tsudik, A practical and provably secure coalition-resistant group signature scheme, (M. Bellare, ed.), Advances in cryptology - crypto 2000, Lecture Notes in Computer Science, vol. 1880, Springer, 2000, p. 255–270.

[And01] R. J. Anderson, *Security Engineering : A Guide to Building Dependable Distributed Systems,* John Wiley & Sons, Inc. New York, 2001.

[AOV96] A. J. Menezes, P. C. v. Oorschot and S. A. Vanstone , *Handbook of Applied Cryptography* , CRC Press LLC, USA, 1996.

[BBS04] D. Boneh, X. Boyen & H. Shacham, Short group signatures, (M. K. Franklin ed.) Advances in Cryptology - Crypto 04, Lecture Notes in Computer Science, vol. 3152, Springer-Verlag, 2004, p. 41–55.

[BMW03] M. Bellare, D. Micciancio & B. Warinschi, Foundations of group signatures : Formal definitions, simplified requirements, and a construction based on general assumptions, (E. Biham ed.), Advances in cryptology - eurocrypt 2003, proceedings, Lecture Notes in Computer Science, vol. 2656, Springer, 2003, p. 614–629.

[Bon98] D. Boneh, The Decision Diffie-Hellman Problem, in J. P. Buhler, ed., Proc. of ANTS III, vol. 1423 of LNCS, pp.48-63.Springer-Verlag, Berlin, Portland, Oregon, June 1998 Stanford University .

[Bre02] E. Bresson, Protocoles cryptographiques pour l'authentification et l'anonymat dans les groupes, Phd thesis, Ecole polytechnique, ENS, Paris, 2002.

[BSS02] E. Bresson, J. Stern, M. Szydlo, Threshold ring signatures and applications to ad-hoc groups, in CRYPTO'02, LNCS 2442, p. 465–480, Springer-Verlag.

[BSZ05] M. Bellare, H. Shi & C. Zhang, Foundations of group signatures : The case of dynamic groups, in CT-RSA'05 (A. Menezes, ed.), Lecture Notes in Computer Science, vol. 3376, Springer, 2005, p. 136–153.

[Cat07] J. Cathalo, Provable Security and Fairness in Cryptographic Identification and Signature Schemes, Phd thesis, Université Catholique de Louvain, Belgique, 2007.

[CG02] S. Canard, M. Girault, Implementing Group Signature Schemes With Smart Cards, in Proceedings of the 5th Smart Card Research and Advanced Application Conference, 2002.

[CL02] J. Camenisch, A. Lysyanskaya, A Signature Scheme with Efficient Protocols, in SCN'02 (S. Cimato, C. Galdi & G. Persiano, eds.), Lecture Notes in Computer Science, vol. 2576, Springer, 2002, p. 268–289

[CL04] J. Camenisch, A. Lysyanskaya, Signature schemes and anonymous credentials from bilinear maps, (M. K. Franklin ed.), Advances in Cryptology - Crypto 04, Lecture Notes in Computer Science, vol. 3152, Springer-Verlag, 2004, p. 56–72.

[Cor03] V. Cortier, Vérification automatique des protocoles cryptographiques, Phd thesis, ENS de Cachan, 2003.

[CP94] L. Chen, T. P. Pedersen, New group signature schemes (extended abstract), in EUROCRYPT'94 (A. D. Santis, ed.), Lecture Notes in Computer Science, vol. 950, Springer-Verlag, 1994, p. 171–181.

[CS97] J. Camenisch , M. Stadler, Efficient group signature schemes for large groups (extended abstract), (B. S. K. Jr. ed.), Advances in Cryptology - Crypto '97, Lecture Notes in Computer Science, vol. 1294, Springer-Verlag, 1997, p. 410–424.

[CSST05] S. Canard, B. Schoenmakers, M. Stam, J. Traoré, List Signature Schemes, Discrete Applied Mathematics, Elsevier, 2005.

[CZMS06] X. Chen, F. Zhang, Y. Mu, W. Susilo , Efficient provably secure restrictive partially blind signatures from bilinear pairings, in Financial cryptography and data security, fc 2006, proceedings, (G. D. Crescenzo & A. Rubin, eds.), Lecture Notes in Computer Science, Springer, 2006, p. 251–265.

[DH76] W. Diffie and M. E. Hellman, New Directions in Cryptography , in IEEE Transactions on Information Theory, vol. 22, 1976, p. 644–654.

[Fou01] P.A. Fouque, Le partage de clés cryptographiques : Théorie et Pratique, Phd thesis, ENS de Paris. 2001

[Fun05] K. T. Fung, *Network Security Technologies*, 2nd Ed, Auerbach Publications, CRC Press LLC, USA, 2005.

[He06] J. Z. D. He, A New Type of Group Blind Signature Scheme Based on Bilinear Pairings, 2006.

[Hoh06] S. Hohenberger, Advances in Signatures, Encryption, and E-Cash from Bilinear Groups, Phd Thesis, Massachusetts Institute Of Technology, 2006

[JLO97] A. Juels, M. Luby, R. Ostrovsky , Security of blind digital signatures (extended abstract), in CRYPTO'97(B. S. K. Jr. ed.), Advances in Cryptology - Crypto '97, Lecture Notes in Computer Science, vol. 1294, Springer-Verlag, 1997, p. 150–164.

[KP97] J. Kilian, E. Petrank, Identity Escrow, 1997.

[KZ06] A. Kiayias & H.-S. Zhou, Concurrent blind signatures without random oracles, in SCN'06 (R. D. Prisco & M. Yung, eds.), Lecture Notes in Computer Science, vol. 4116, Springer, 2006, p. 49–62.

[LW03] J. Lv, X. Wang, Verifiable ring signature, in CANS'03, Proc. Of DMS 2003, 2003, p. 663–665.

[LWW04] J. K. Liu, V. K. Wei, D. S. Wong, Linkable Spontaneous Anonymous Group Signature for Ad Hoc Groups, in ACISP'04 (H. Wang, J. Pieprzyk & V. Varadharajan, eds.), Lecture Notes in Computer Science, vol. 3108, p. 325–335, Springer, 2004.

[Lys02] A.Lysyanskaya, Signature Schemes and Applications to Cryptographic Protocol Design, Phd thesis, Massachusetts Institute of Technology, 2002.

[Men07] N. Mentens, Secure and Efficient Coprocessor Design for Cryptographic Applications on FPGAs, Phd thesis, Université Catholique de Louvain, Belgique, 2007.

[Mol07] R. A. Mollin, *An Introduction To Cryptography*, 2nd Ed, (Discrete Mathematics And Its Applications, Series Editor Kenneth H. Rosen), Chapman & Hall/CRC,Taylor & Francis Group, USA, 2007.

[Oka06] T. Okamoto, Efficient blind and partially blind signatures without random oracles, in TCC'06, (S. Halevi & T. Rabin eds.), Theory of cryptography, tcc 2006, proceedings, Lecture Notes in Computer Science, vol. 3876, Springer, 2006, p. 80–99

[PBO+04] B. Preneel, A. Biryukov, S. B. ¨Ors, C. De Cannière, E. Oswald, B. Van Rompay, L. Granboulan, E. Dottax, G. Martinet, S. Murphy, A. Dent, R. Shipsey, C. Swart, J. White, M. Dichtl, S. Pyka, M. Schafheutle, P. Serf, E. Biham, E. Barkan, Y. Braziler, O. Dunkelman, V. Furman, D. Kenigsberg, J. Stolin, J-J. Quisquater, M. Ciet, F. Sica, H. Raddum, L.Knudsen, M. Parker, New European Schemes for Signatures, Integrity and Encryption, Final report of European project number IST-1999-12324, Nessie, Springer-Verlag, 2004.

[Pet97] H. Petersen, How to convert any digital signature scheme into a group signature scheme, in Security Protocols Workshop (B. Christianson, B. Crispo, T. M. A. Lomas & M. Roe, eds.), Lecture Notes in Computer Science, vol. 1361, Springer, 1997, p. 177–190.

[Pop03] C. Popescu, A secure and efficient groupe blind signature scheme, Studies in Informatics and Control, Vol.12, No.4 , 2003.

[PS96] D. Pointcheval, J. Stern, Provably secure blind signature schemes, (K. Kim & T. Matsumoto eds.), Advances in Cryptology - Asiacrypt '96, Proceedings, Lecture Notes in Computer Science, vol. 1163, Springer-Verlag, 1996, p. 252–265.

[Ram99] Z. A. Ramzan, Group Blind Digital Signatures: Theory and Applications, thèse de master de science, Massachusetts institute of technology, 1999.

[RSA78] R. L. Rivest, A. Shamir & L. M. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Commun. ACM 21 (1978), no. 2, p. 120–126.

[RST01] R. L. Rivest, A. Shamir, Y. Tauman , How to leak a secret : Theory and Applications of Ring Signatures, in ASIACRYPT'01, (C. Boyd ed.), Advances in Cryptology, Lecture Notes in Computer Science, vol. 2248, Springer-Verlag, 2001, p. 552–565.

[SPC95] M. Stadler, J.M Piveteau, J. Camenisch, Fair Blind Signatures, In: Advances in Cryptology - Eurocrypt'95, volume 921 of Lecture Notes in Computer Science, pp. 209-219, Springer-Verlag, 1995, p. 209–219.

[Tan07] Q. Tang, Key Establishment Protocols and Timed-Release Encryption Schemes, Phd thesis, Royal Holloway University of London, 2007.

[TW05] P. P. Tsang, V. K. Wei, Short Linkable Ring Signatures for E-voting, E-cash and Attestation, ISPEC, 2005

[TWC+04] P. P. Tsang, V. K. Wei1, T. K. Chan, M. H. Au, J. K. Liu, D. S. Wong, Separable Linkable Threshold Ring Signatures, (A. Canteaut and K. Viswanathan. Eds.): INDOCRYPT 2004, Lecture Notes in Computer Science vol 3348, pp. 384–398, 2004.

[ZK02] F. Zhang & K. Kim, Id-based blind signature and ring signature from pairings, in ASIACRYPT'02 (Y. Zheng ed.)Advances in Cryptology - Asiacrypt '02, Proceedings, Lecture Notes in Computer Science, vol. 2501, Springer-Verlag, 2002, p. 533–547.

[ZWW05] J. Zhang, Q. Wu, Y. Wang, A New Efficient Group Signature With Forward Security, in Informatica 29 (2005) 321–325, 2005.