Université Cheikh Anta DIOP de Dakar

Faculté des Sciences et Techniques

THESE DE DOCTORAT UNIQUE ÈS SCIENCES MATHÉMATIQUES

Spécialité :

Codage, Cryptologie, Algèbre et Applications

TITRE :

Courbes elliptiques, Cryptographie à clés publiques et Protocoles d'échange de clés

présenté en vue de l'obtention du grade de
Docteur de l'Ecole Doctorale de Mathématiques, Informatique et Sciences Connexes

par

Demba SOW

Soutenu le 06 / juillet / 2013 devant le jury :

Prénoms et Nom	***Qualité***	***Université***
Pr. Mamadou SANGHARE	Président	Cheikh Anta Diop (Dakar)
Pr. Abdelhak AZHARI	Rapporteur	Hassan II (Casablanca /Maroc)
Dr. Youssouf Mahamadou DIAGANA	Rapporteur	Abobo-Adjamé (Cote d'ivoire)
Pr. Cheikh Thiécoumba GUEYE	Examinateur	Cheikh Anta Diop (Dakar)
Dr. Oumar DIANKHA	Examinateur	Cheikh Anta Diop (Dakar)
Pr. Djiby SOW	Directeur de thèse	Cheikh Anta Diop (Dakar)

Année académique 2012-2013

DEDICACES

Qu'ALLAH soit Loué !

Je rends grâce à DIEU

PAIX et SALUT sur le PROPHETE MOHAMED

je dédie cette thèse à :

- ✠ mes parents : ma mère Farmata MBODJI et mon père Ibrahima SOW
- ✠ ma femme Awa SOW
- ✠ ma fille Yaram SOW
- ✠ mes frères et soeurs
- ✠ mes homonymes
- ✠ mes cousins et cousines
- ✠ mes neveux et nièces
- ✠ mes oncles et tantes
- ✠ mes amis et camarades de tous horizons
- ✠ mes collègues
- ✠ mes maîtres et professeurs
- ✠ mes élèves et étudiants
- ✠ tous ceux qui m'ont soutenu de près ou de loin
- ✠ toutes ces personnes qui m'ont tout donné sans rien demander

■ *Je dédie plus particulièrement cette thèse à un ami qui m'est très cher et qui nous a quitté le lundi 03 juin 2013 ; je veux nommer* ***Djibril SECK****. Que la terre de Ndiaw lui soit légère et que le Bon DIEU l'accueille dans son paradis, AMIIN !!*

REMERCIEMENTS

Je remercie ALLAH Le Tout Puissant

Le Clément et Le Miséricordieux

Paix et Salut sur le Prophète

✠ Je tiens à remercier vivement Monsieur le Professeur **Mamadou SANGHARE** pour l'honneur qu'il m'a fait en acceptant de présider mon jury de thèse. Je le remercie également pour l'honneur qu'il m'a fait en me confiant l'organisation du séminaire du Laboratoire d'Algèbre, de Cryptologie, de Géométrie Algébrique et Applications. Je le félicite d'avoir pu créer un bon laboratoire de recherche.

✠ Je remercie très sincèrement les **Pr. Abdelhak AZHARI** et **Dr. Youssouf Mahamadou DIAGANA** pour avoir accepté d'être rapporteurs de cette thèse et examinateurs dans ce jury.

✠ J'exprime ma profonde gratitude aux **Pr. Cheikh Thiécoumba GUEYE** et **Dr. Oumar DIANKHA** pour l'honneur qu'ils me font en acceptant de siéger dans le jury de cette thèse.

✠ Je tiens à remercier sincèrement le Profeseur **Djiby SOW** qui a dirigé mes travaux. Je lui exprime ma profonde gratitude pour tout ce qu'il a fait pour moi et que je ne saurais entièrement lister ici. Particulièrement, je le remercie très chaleureusement pour l'excellente formation que j'ai reçue auprès de lui sur tous les plans (didactique, recherche, humain,..). J'ai apprécié tout particulièrement sa patience, sa capacité d'écoute et sa grande disponibilité à mon égard.

✠ Je remercie :

- ⊛ tous les membres du Laboratoire d'Algèbre, de Cryptologie, de Géométrie Algébrique et Applications (LACGAA),
- ⊛ mes compagnons de tous les jours à la fac, je veux nommer Mamadou G. CAMARA, Régis B., Youssef, EL. Modou MBOUP, Chérif Bachir DEME, Lamine DIOUF, Ahmed Khalifa, Abou Aziz CISS, Amadou TALL, Landing FALL, Raoul TSIBA, André MIALEBAMA.
- ⊛ tous mes collègues et tout le personnel administratif et technique du Département de Mathématiques et Informatique et de la Faculté des Sciences et Techniques de l'Université Cheikh Anta Diop de Dakar.
- ⊛ tous mes étudiants en Licence et Master TDSI, en L1PCSM, en Licences (1, 2 et 3) ESSA, à l'ESMT, à l'ISI et à l'ESP ;

Table des matières

Introduction :

Cryptographie moderne :

Le rôle de la cryptographie est de garantir la sécurité des communications c'est dire de permettre à des entités qui ne se font pas confiance en général de communiquer en toute sécurité en présence de potentiels adversaires (susceptibles entre autres d'intercepter et de modifier les informations échangées ou d'usurper des identités). Mais la cryptographie à elle seule ne prend pas en charge tous les besoins de sécurité et n'est donc pas la seule discipline qui intervienne dans la sécurité des communications. La cryptographie est composée de la cryptographie symétrique et de la cryptographie asymétrique.

La cryptographie symétrique consiste à échanger des données chiffrées à partir d'un secret (appelé clé) connu uniquement par les parties concernées, le chiffrement comme le déchiffrement nécessitant la clé. Pour cela, il paraît nécessaire d'avoir un protocole d'échange de clés sûr et efficace.

De l'antiquité jusque dans les années 1970, tous les systèmes de cryptographie étaient symétriques (donc la principale préoccupation était la confidentialité) et il n'existait pas de méthode sûre pour échanger les clés secrètes.

Historiquement la cryptographie symétrique est le premier type de chiffrement utilisé et elle fournit le seul chiffrement théoriquement indéchiffrable (chiffrement de Vernam ou one-time pad) d'après la théorie de Shannon (1949). Elle est d'une grande efficacité en terme de temps de calculs.

C'est en 1976 que deux chercheurs White Diffie et Martin Hellmann [11] à l'université de Standford, dans leur papier "New Direction In Cryptography", ont donné une solution au problème de l'échange des clés et ont proposé en même temps le modèle théorique de la cryptographie à clé publique qui est appelé modèle de cryptographie asymétrique.

L'une des plus grandes avancées en matière de cryptographie à clé publique depuis son apparition est la méthodologie de la sécurité prouvée, qui complète la cryptanalyse en garantissant en quelque sorte l'absence de failles. Il s'agit dans un premier temps de modéliser la notion même de sécurité, puis de construire des cryptosystèmes prouvés sûrs dans ce modèle, sous des hypothèses mathématiques précises et plausibles. Mais il convient de relativiser la portée des résultats obtenus. Aujourd'hui, la "bonne" notion de sécurité communément admise est ce qu'on appelle l'indistinguabilité sous des attaques adaptatives à chiffrés choisis pour le chiffrement à clé publique, et la résistance aux contrefaçons existentielles sous des attaques à messages choisis pour la signature numérique.

Protocole d'échange de clés :

L'échange de clé de Diffie-Hellman a été développé par ces deux auteurs en 1976 et publié dans l'article : W. Diffie and M.E. Hellman, *New directions in cryptography*, IEEE Transactions on Information Theory 22 (1976), 644-654. Ce protocole de Diffie-Hellman est le premier protocole d'échange de clés.

Nous rappelons que l'échange d'une clé secrète est fondamental en cryptographie. En effet tout chiffrement d'une grande quantité de données ne peut se faire qu'avec du chiffrement à clé secrète, surtout si cet échange a lieu en temps réel, en raison de la lenteur relative des chiffrements à clé publique.

Le développement rapide des réseaux de communication numériques, ainsi que l'augmentation du nombre des utilisateurs, ont posé de façon de plus en plus critique le problème de l'échange des clés de chiffrement.

Courbes elliptiques :

Si le principe de la cryptographie à clé publique fut proposé en 1975, ce n'est qu'en 1977 que fut présenté le premier protocole effectif : RSA (du nom de ses auteurs Rivest, Shamir et Adleman). Principalement basé sur le problème de la factorisation des grands entiers, RSA est encore aujourd'hui la primitive la plus utilisée en cryptographie. Cependant les nombreux progrès effectués dans le domaine de la factorisation font que la taille des clés RSA augmente plus vite que ne le requiert l'augmentation de la puissance des ordinateurs. Donc, il faut des schémas qui réduisent la taille des clés. C'est l'une des raisons pour lesquelles la cryptographie basée sur les courbes elliptiques (ECC) connaît un tel intérêt depuis son introduction par Miller et Koblitz [25, 32] en 1987.

Les cryptosystèmes sur les courbes elliptiques sont basés sur le problème du logarithme discret dans un sous-groupe cyclique du groupe de points rationnels sur une courbe elliptique définie sur un corps fini. Ainsi la sécurité de tous les cryptosystèmes et protocoles sur les courbes elliptiques est basée sur l'insolubilité apparente du problème du logarithme discret des courbes elliptiques (ECDLP).

Pourquoi utilise-t-on les courbes elliptiques en cryptographie ?

- Il n'existe pas d'algorithme sous-exponentiel en temps connu pour la ECDLP et donc les paramètres de petite taille peuvent être sélectionnés pour les cryptosystèmes sur les courbes elliptiques et on atteint le même niveau de sécurité que pour les schémas ordinaires du logarithme discret sur les corps finis ou pour RSA.
- ECC requiert à niveau de sécurité équivalent, des clés bien plus petites que RSA (une clé ECC de 160 bits est aussi robuste qu'une clé RSA de 1024 bits), celui-là étant donc plus adapté à des environnements à puissance réduite (tels que les cartes à puce).

Applications en cryptographie : Les courbes elliptiques sont utilisées :

- dans les algorithmes de factorisation d'entiers ;

– dans les algorithmes prouvant la primalité ;
– pour l'échange des clés Diffie Hellmann ;
– pour la conception de systèmes à clé publique : le chiffrement et les signatures (par exemple le cryptosystème El Gamal) ;
– pour l'extraction aléatoire et la conception des générateurs de nombres pseudo-aléatoires cryptographiquement sûrs.

Jusqu'au milieu des années 90, les principaux travaux concernant ECC ont porté sur l'amélioration de son efficacité en termes de temps de calcul et de ressources, ainsi que sur la robustesse du problème du logarithme discret. Cependant, en 1996, Paul Kocher présenta une nouvelle forme d'attaque dite par canaux cachés [27]. Le principe n'est pas de résoudre le problème théorique sur lequel repose la sécurité du protocole cryptographique, mais de tirer partie des informations que relâche immanquablement le système physique sur lequel est implanté le protocole (chaleur, consommation d'énergie, temps de calcul etc) afin de retrouver la clé secrète du système.

Les protocoles cryptographiques doivent s'exécuter le plus rapidement possible afin de ne pas ralentir les échanges de données. Toutefois, il faut veiller à ce que l'implantation, aussi bien logicielle que matérielle, soit la moins sujette possible aux attaques par canaux cachés. Ce travail d'implantation peut s'effectuer à deux niveaux : au niveau de l'arithmétique de la courbe et au niveau de l'arithmétique du corps sous-jacent.

Cela a motivé la découverte et l'étude de nouveaux algorithmes de calcul rapide et de nouvelles formes de courbes elliptiques telles que les courbes d'Edwards, les courbes de Huff, les courbes Essiens etc ...

Contribution de la thèse :

Dans le cadre de cette thèse, nous apportons quelques contributions à la cryptographie asymétrique et plus particulièrement au cryptosystème El Gamal, à l'échange de clé Diffie-Hellman et aux courbes elliptiques. Notre contribution est composée des chapitres 4, 5 et 6 :

1) Nous avons présenté de nouveaux schémas de chiffrement et de signature généralisant ceux d'El Gamal en proposant beaucoup plus de variantes et en améliorant (légèrement) le processus de déchiffrement en terme de rapidité.
2) Nous avons proposé une cryptanalyse et une amélioration en terme de sécurité du protocole d'échange de clés "Strong Diffie-Hellman-DSA Key Exchange" basé sur l'algorithme de signature DSA [14].
3) Nous avons étudié de nouvelles formes de courbes elliptiques à l'image des courbes de Huff et d'Edwards et prouvé l'équivalence birationnelle de certaines courbes elliptiques à celles que nous avons proposées.

Organisation du mémoire :

Le présent mémoire est composé de six chapitres dont trois sur les notions de bases et trois pour la partie contribution.

L'organisation des six chapitres est la suivante :

Chapitre 1 **:** Il est destiné aux notions de base sur les courbes elliptiques et est composé de trois sections à savoir généralités sur les courbes elliptiques, les courbes de Huff non binaires et les courbes de Huff binaires à titre d'illustration des nouvelles formes de courbes elliptiques étudiées récemment et différentes de la forme de Weierstrass.

Chapitre 2 **:** On a fait un survol sur la cryptographie à clé secrète et la cryptographie à clé publique et les preuves de sécurité. Dans cette partie, nous avons parlé des systèmes de cryptographie et de la sécurité des schémas de signature et de chiffrement.

Chapitre 3 **:** Ici nous avons abordé quelques notions sur les protocoles : généralités sur les protocoles, protocoles cryptographiques et présentation de quelques protocoles d'échange de clés.

Chapitre 4 **:** Dans ce chapitre, nous avons commencé par une présentation d'une nouvelle généralisation des schémas de chiffrement et de signatures El Gamal en énonçant les algorithmes de génération des clés, de chiffrement et de déchiffrement mais également ceux de signature et de vérification avant d'étudier la sécurité du problème lié au générateur publique et de mettre en évidence l'apport de ce nouveau cryptosystème.

Chapitre 5 **:** Dans ce chapitre, nous avons effectué une attaque sur le protocole "Strong Diffie-Hellman-DSA Key Exchange" de Jeong et *al.* [23] et nous avons proposé un nouveau protocole d'échange dénommé "Strong Diffie-Hellman-Exponential-Schnnor Key Exchange" basé sur l'exponentiation modifié de Schnorr et qui vient améliorer les insuffisances du protocole d'échange de Jeong et *al.*, et contrairement au protocole de Jeong et *al.*, ici une preuve de sécurité complète est proposée.

Chapitre 6 **:** Dans ce chapitre, comme les cryptosystèmes El Gamal et l'échange de clés Diffie-Hellman ont une meilleure sécurité sur les courbes elliptiques, nous avons proposé de nouvelles formes de courbes elliptiques. Nous avons prouvé que la plupart des courbes elliptiques sont birationnellement équivalentes à l'une des courbes que nous avons proposées. A l'étape actuelle dans nos recherches, nous n'avons pas encore pu montrer que la loi d'addition sur cette nouvelle courbe est unifiée et/ou complète. Donc, les courbes que nous avons proposées ne sont pas encore utilisables en l'état, en cryptographie.

Notations :

A travers tout le document :

- $a \longleftarrow b$ signifie que le contenu de b est affecté à a ;
- $x \overset{Rand}{\longleftarrow} G$ signifie que x est aléatoirement choisi dans G ;
- $\{0,1\}^n$ est l'ensemble des chaines de longueur n bits ;
- $\{0,1\}^*$ est l'ensemble des chaines bits possibles.

Chapitre 1

Courbes elliptiques

1.1 Généralités sur les courbes elliptiques

Introduites par Miller et Koblitz [25, 32] en 1987, les courbes elliptiques connaissent un essor considérable depuis leur utilisation dans la cryptographie. Reposant sur le problème du logarithme discret, la cryptographie sur les courbes elliptiques appliquées requiert, à niveau de sécurité équivalent, des clés bien plus petites que RSA (une clé de 160 bits est aussi robuste qu'une clé RSA de 1024 bits), celui-là étant donc plus adapté à des environnements à puissance réduite (tels que les cartes à puce). Pour de plus amples informations voir les livres et les articles suivants :

- Silverman [42]
- Hand book hyper elliptic curves[3]
- Courbes d'Edwards (Edwards curves) [16, 5]
- Huff curves [22, 10]

1.1.1 Notions de base

Espace affine

Soit K un corps et $\overline{K}$ sa clôture algébrique. L'espace affine de dimension n est l'ensemble des points $\mathbb{A}^n(\overline{K}) = \{P = (a_1, \cdots, a_n),\ a_i \in \overline{K}\}$.

L'ensemble des points rationnels est $\mathbb{A}^n(K) = \{P = (a_1, \cdots, a_n),\ a_i \in K\}$.

Ensemble algébrique affine

On note $\overline{K}[X_1, \cdots, X_n]$ l'anneau des polynômes à n variables à coefficients dans $\overline{K}$.

– Si I est un idéal de $\overline{K}[X_1, \cdots, X_n]$, alors le sous-ensemble de $\mathbb{A}^n(\overline{K})$ donné par

$$\mathcal{C}_I = \{P \in \mathbb{A}^n \ / f(P) = 0, \forall f \in I\}$$

est appelé ensemble algébrique défini par I.

– Si $\mathcal{C} \subseteq \mathbb{A}^n(\overline{K})$ est un ensemble algébrique affine, son idéal est

$$I(\mathcal{C}) = \{f \in \overline{K}(X_1, \cdots, X_n) \ / f(P) = 0, \forall P \in \mathcal{C}\}.$$

– On dit qu'un ensemble algébrique $\mathcal{C}$ est défini sur K si son idéal est engendré par des polynômes de $K[X]$.
– Un ensemble algébrique $\mathcal{C}$ est une variété si son idéal $I(\mathcal{C})$ est premier.
– Si $\mathcal{C}$ est une variété alors l'anneau des coordonnées de $\mathcal{C}$ est $\overline{K}[\mathcal{C}] = \frac{\overline{K}[X_1,\cdots,X_n]}{I(\mathcal{C})}$
– L'anneau des coordonnées $\overline{K}[\mathcal{C}]$ est intègre et son corps des fractions, noté $\overline{K}(\mathcal{C})$, est appelé le corps des fractions de $\mathcal{C}$.
– La dimension d'une variété $\mathcal{C}$, est le degré de transcendance de l'extension de corps de $\overline{K}(\mathcal{C})$ sur $\overline{K}$.
– En particulier si $\mathcal{C}$ est donné par $H(X_1, \cdots, X_n) = 0$ avec $H(X_1, \cdots, X_n)$ un polynôme irréductible dans $\overline{K}$ (c'est-à-dire absolument irréductible dans K), alors $\mathcal{C}$ est une variété affine de dimension $n-1$.

NB :

Dans la suite on considère des variétés affines de dimension 1 en définissant une variété $\mathcal{C}$ par $\mathcal{C} : \ H(x, y) = 0$ où $H(x, y)$ est polynôme irréductible dans $\overline{K}$.

Espace projectif

Un espace projectif sur K, noté $\mathbb{P}^n$ ou $\mathbb{P}^n(\overline{K})$, est l'ensemble des points $(n+1)-$uplets

$$(X_0, \cdots, X_n) \in \mathbb{A}^{n+1}$$

tel qu'au moins un X_i est non nul, modulo la relation d'équivalence donnée par

$$(X_0, \cdots, X_n) \sim (Y_0, \cdots, Y_n)$$

s'il existe un $\lambda \in \overline{K}^*$ avec $X_i = \lambda Y_i$ pour tout i. Une classe d'équivalence $\{(\lambda X_0, \cdots, \lambda X_n), \ \lambda \in \overline{K}^*\}$ est notée $[X_0, \cdots, X_n]$ et $X_0, \cdots, X_n$ sont appelés coordonnées homogènes pour les points correspondants dans $\mathbb{P}^n$. L'ensemble des points $K-$rationnels dans $\mathbb{P}^n$ est l'ensemble

$$\mathbb{P}^n(K) = \{[X_0, \cdots, X_n] \in \mathbb{P}^n : \text{ pour tout } X_i \in K\}$$

Ensemble algébrique projectif

– Soit $P = [X_0, \cdots, X_n] \in \mathbb{P}^n(\overline{K})$. Le corps minimal de définition pour P (sur K), noté $K(P)$, est le corps

$$K(P) = K(X_0/X_i, \cdots, X_n/X_i) \text{ pour tout } X_i \neq 0.$$

c'est-à-dire la plus petite extension de corps contenant K et les valeurs X_j/X_i, $0 \leq j \leq n$, $X_i \neq 0$.

– Un polynôme $f \in \overline{K}[X] = \overline{K}[X_0, \cdots, X_n]$ est de degré homogène d si

$$f(\lambda X_0, \cdots, \lambda X_n) = \lambda^d f(X_0, \cdots, X_n)$$

pour tout $\lambda \in \overline{K}$. Un idéal $I \subset \overline{K}[X]$ est homogène s'il est généré par des polynômes homogènes.
Pour chaque idéal homogène I, on associe un sous-ensemble de $\mathbb{P}^n$,

$$V_I = \{P \in \mathbb{P}^n : f(P) = 0 \text{ pour tout } f \in I\},\ I \text{ homogène.}$$

– Un ensemble algébrique (projectif) est tout ensemble de la forme V_I. Si V est un ensemble algébrique projectif, l'idéal (homogène) de V, noté $I(V)$, est l'idéal dans $\overline{K}[X]$ donné par

$$I(V) = \{f \in \overline{K}[X] : f \text{ est homogène et } f(P) = 0 \text{ pour tout } P \in V\}.$$

Points infinis

A toute variété affine W, on peut associer une et une seule variété projective V. Les points infinis de W sont les points de V qui ne sont pas sur W.

Exemples

– Si on a une variété affine donnée par $H(x, y) = 0$, on peut avoir la version projective en faisant le changement : $x = \dfrac{X}{Z}$, $y = \dfrac{Y}{Z}$ puis réduire en supprimant les dénominateurs.
– Si on a une variété projective $H(X, Y, Z) = 0$, on peut avoir la version affine en posant $Z = 1$.
– Pour trouver les points infinis, il suffit de résoudre $H(X, Y, Z) = 0$ en posant $Z = 0$

Singularité

Une courbe $\mathcal{C}$ de dimension 1 sur K admet une partie affine qui peut être représentée comme un polynôme $H(x, y)$ sur $K[x, y]$ et alors, la partie affine de la courbe devient $H(x, y) = 0$. Si la courbe "affine" est définie par $H(x, y) = 0$, alors :
la courbe $\mathcal{C}$ est non singulière ou lisse signifie que le système :

$$\begin{cases} H(x,y) = 0, & (1)\,; \\ \dfrac{\partial H(x,y)}{\partial x} = 0, & (2)\,; \\ \dfrac{\partial H(x,y)}{\partial y} = 0, & (3). \end{cases}$$

n'a pas de solutions dans K. Le résultat se généralise aisément à n variables.

Genre

Soit $\mathcal{C}$ une courbe non-singulière sur un corps K, une équation de Weierstrass de $\mathcal{C}$ est un modèle affine donné par la forme :

$$H(x, y) = y^2 + yh(x) - f(x) = 0$$

où f, h sont des polynômes dans $K[x]$, f est unitaire et $deg(h) \leq \lfloor (deg(f) - 1)/2 \rfloor$ où $\lfloor a \rfloor$ = partie entière de a.

Notez que si $\deg(f) = 2g + 1$ ou $\deg(f) = 2g + 2$, alors $\deg(h) \leq g$, g est un entier naturel non nul.

L'entier g est unique et est un invariant de la courbe $\mathcal{C}$. Dans le cas général, il est donnée par le théorème de Riemann-Roch et est appelé genre de la courbe (voir Silvermann [42], Hank Book [3]).

Pour une courbe elliptique en forme de Weierstrass, on a $h(x) = a_1x + a_3$ et $f(x) = x^3 + a_2x^2 + a_4x + a_6$ et donc le genre est $g = 1$.

Points rationnels

L'ensemble des points $K-$rationnels de la courbe $\mathcal{C}$ est défini par l'ensemble des points :
$\mathcal{C}(K) = \{(x, y) \in K \times K, /H(x, y) = 0\} \cup S_\infty$ où S_∞ est l'ensemble des points infinis.

Equivalence birationnelle

Pour étudier les isomorphismes entre les courbes et plus généralement entre les variétés, nous avons besoin d'une fonction birationnelle que nous utiliserons.

Avant de la définir, nous rappelons qu'un morphisme entre deux variétés est une fonction qui est compatible avec la structure de variété.

Soient A et B deux variétés sur un corps K, un morphisme birationnel de A à B est un morphisme $\psi : A \longrightarrow B$ avec un sous ensemble ouvert U dense dans B tel que $\psi^{-1}(U) \longrightarrow U$ est un isomorphisme (voir Silverman [42]).

1.1.2 Premières propriétés et lois de groupe

Définition 1.1.1. *Une courbe projective définie sur un corps K qui est non singulière et irréductible sur la clôture algébrique $\overline{K}$ de K, de genre 1 et qui a au moins un point $K-$rationnel, est appelée courbe elliptique sur K.*

Définition 1.1.2. .
Une courbe elliptique E sur un corps K notée par E/K est donnée par l'équation de Weierstrass

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \qquad (1)$$

où les coefficients $a_1, a_2, a_3, a_4, a_6 \in K$ sont tels que pour tout point (x_1, y_1) avec des coordonnées dans K satisfaisant (1), les dérivées partielles $2y_1 + a_1x_1 + a_3$ et $3x_1^2 + 2a_2x_1 + a_4 - a_1y_1$ ne s'annulent pas simultanément.

La dernière condition indique qu'une courbe elliptique est non-singulière (c'est-à-dire lisse). Un point, sur une courbe, est dit singulier si les deux dérivés partielles s'annulent. Pour une représentation plus courte, nous regroupons les coefficients dans (1) sous la forme suivante :

$E : y^2 + h(x)y = f(x)$, *et les données $h(x)$ et $f(x) \in K[x]$, $deg(h) \leq 1$, $deg(f) = 3$ avec f unitaire.*

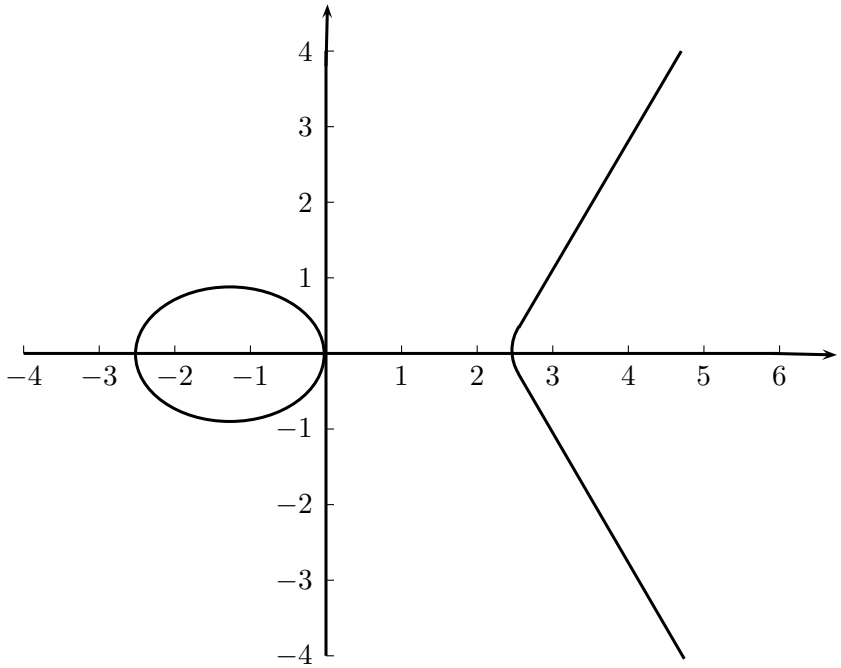
La régularité de la condition peut aussi être exprimée plus intrinsèquement. En effet, soit

$$b_2 = a_1^2 + 4a_2, \qquad b_4 = a_1a_3 + 2a_4,$$
$$b_6 = a_3^2 + 4a_6, \qquad b_8 = a_1^2a_6 - a_1a_3a_4 + 4a_2a_6 + a_2a_3^2 - a_4^2$$

En caractéristique impaire, la transformation $y \mapsto y - (a_1x + a_3)/2$ mène à une courbe isomorphique donnée par

$$y^2 = x^3 + \frac{b_2}{4}x^2 + \frac{b_4}{2}x + \frac{b_6}{4} \qquad (2).$$

Exemple d'un schéma d'une courbe elliptique sur $\mathbb{R}$: $y^2 = x^3 - 6x$



Le polynôme cubique précédent admet seulement des racines simples sur la clôture algébrique $\overline{K}$ si et seulement si son déterminant est non nul. L'équation du déterminant est par conséquent utile pour déterminer si (2) est une courbe elliptique ou non. En plus, il est également approprié pour les corps de caractéristique 2.

Définition 1.1.3. *Soit E une courbe définie sur K par (1) et soient les coefficients b_2, b_4, b_6 et b_8 définis comme précédemment. Le discriminant de la courbe E noté par Δ satisfait*

$$\Delta = -b_2^2b_8 - 8b_4^3 - 27b_6^2 + 9b_2b_4b_6.$$

La courbe E est non-singulière , et ainsi est une courbe elliptique, si et seulement si Δ est non nul. Dans ce cas, nous définissons le $j-$invariant de E par $j(E) = (b_2^2 - 24b_4)^3/\Delta$.

Le $j-$invariant permet de classer les courbes par classe d'isomorphismes.

Pour additionner deux points $P = (x_1, y_1)$ et $Q = (x_2, y_2)$, en général on trace une droite les reliant. Il existe un troisième point d'intersection. Le symétrique de ce point par rapport à l'axe des abscisses donne la somme de P et Q notée $P \oplus Q$. La même construction peut être appliquée

à un point double où la droite qui relie les points est remplacée par la tangente à P. On appelle le conjugué d'un point P, son symétrique par rapport à l'axe des abscisses.

Soient $P = (x_1, y_1)$ et $Q = (x_2, y_2)$ deux points d'une courbe elliptique E, alors :

- si $P = (x_1, y_1)$ et $Q = (x_2, y_2)$ sont distincts et non conjugués (c'est à dire $P \neq Q$ et $P \neq \overline{Q}$) alors la droite (PQ) coupe la courbe E en un unique troisième point $R = (x_3, y_3)$ et la somme de $P = (x_1, y_1)$ et $Q = (x_2, y_2)$ est le conjugué de $R = (x_3, y_3)$: $P \oplus Q = \overline{R}$.

- Si $P = Q$, on considère la droite tangente en P à la courbe E au lieu de la droite (PQ).

Supposons $P \neq Q$ avec $(x_1 \neq x_2)$ comme précédemment et calculons les coordonnées de $\overline{R} = P \oplus Q = (x_3, y_3)$. La droite d'intersection admet comme pente

$$\lambda = \frac{y_1 - y_2}{x_1 - x_2}$$

et passe à travers P. Son équation est ainsi donnée par :

$$y = \lambda x + \frac{x_1 y_2 - x_2 y_1}{x_1 - x_2}.$$

Nous appelons le terme constant par μ et remarquons que $\mu = y_1 - \lambda x_1$. Les points d'intersection avec la courbe sont obtenus en égalisant la droite et la courbe :

$$(\lambda x + \mu)^2 + (a_1 x + a_3)(\lambda x + \mu) = x^3 + a_2 x^2 + a_4 x + a_6.$$

Ceci conduit à l'équation $r(x) = 0$ où

$$r(x) = x^3 + (a_2 - \lambda^2 - a_1\lambda)x^2 + (a_4 - 2\lambda\mu - a_3\lambda - a_1\mu)x + a_6 - \mu^2 - a_3\mu.$$

Nous connaissons déjà deux racines de $r(x)$, correspondant aux abscisses des deux autres points. Puisque

$$r(x) = (x - x_1)(x - x_2)(x - x_3)$$

on a $\lambda^2 + a_1\lambda - a_2 = x_1 + x_2 + x_3$. Comme x_1, x_2 sont définis sur K alors il existe de même x_3 et $y_3 = \lambda x_3 + \mu$. L'inflexion à l'axe des abscisses peut être traduite à la condition que le second point ait les mêmes coordonnées et ainsi satisfait à l'équation de la courbe. Nous observons que si $P = (x_1, y_1)$ est sur la courbe alors il en est de même que $(x_1, -y_1 - a_1 x_1 - a_3)$, qui correspond à $-P$, puisque le point à l'infini est l'élément neutre pour cette loi. Par conséquent, nous obtenons $y_3 = -\lambda x_3 - \mu - a_1 x_3 - a_3$.

Le dédoublement de $P = (x_1, y_1)$ fonctionne aussi de même avec la pente obtenue par une dérivation implicite. D'où nous avons $P \oplus Q = (x_3, y_3)$ et $-P = (x_1, -y_1 - a_1 x_1 - a_3)$,

$P \oplus Q = (\lambda^2 + a_1\lambda - a_2 - x_1 - x_2, \lambda(x_1 - x_3) - y1 - a_1 x_3 - a_3)$, où

$$\lambda = \begin{cases} \dfrac{y_1 - y_2}{x_1 - x_2} & \text{si } P \neq \pm Q \\ \\ \dfrac{3x_1^2 + 2a_2 x_1 + a_4 - a_1 y_1}{2y_1 + a_1 x_1 + a_3} & \text{si } P = Q \end{cases}$$

Il découle immédiatement de cette description illustrée que cette loi est commutative, admet le point à l'infini comme élément neutre et que l'inverse de (x_1, y_1) est donné par $(x_1, -y_1 - a_1x_1 - a_3)$. L'associativité beaucoup plus délicate peut être montrée en appliquant seulement la loi de groupe (voir Silverman [42]).

1.1.3 Courbes elliptiques sur $\mathbb{F}_{2^m}$

Nous rappelons des cas particuliers des courbes elliptiques binaires qui nous intéresserons par la suite.

Définition 1.1.4. *Soit $\mathbb{K} = \mathbb{F}_{2^m}$ un corps, une courbe elliptique binaire est une équation de Weierstrass de la forme*

$$\mathbb{E}/_{\mathbb{F}_{2^m}} : \ y^2 + xy = x^3 + a_2x^2 + a_6, \qquad (\textit{avec } a_6 \neq 0)$$

où $\mathcal{O} = (0 : 1 : 0)$ est le point à l'infini et l'inverse d'un point $P_0 = (x_0, y_0) \in \mathbb{E}\backslash\{\mathcal{O}\}$ est $-P_0 = (x_0, y_0 + x_0)$.

Lois de groupe (formules pour le doublement et l'addition) :

Soient $P_1 = (x_1, y_1)$ et $P_2 = (x_2, y_2)$, l'addition de $P_1 + P_2 = P_3 = (x_3, y_3)$ avec

$$\begin{cases} x_3 = \lambda^2 + \lambda + x_1 + x_2 + a_2, \\ \\ y_3 = (x_1 + x_3)\lambda + x_3 + y_1. \end{cases} \qquad \text{où} \qquad \begin{cases} \lambda = \dfrac{y_1 + y_2}{x_1 + x_2} \text{ si } P_1 \neq P_2, \\ \\ \lambda = \dfrac{y_1}{x_1} + x_1 \text{ si } P_1 = P_2. \end{cases}$$

1.1.4 Multiplication scalaire

Supposons $n \in \mathbb{N}\backslash\{0\}$ et notons la multiplication scalaire par n sur E par $[n]$, ou $[n]_E$ pour éviter la confusion. C'est-à-dire,

$$[n] : E \to E$$
$$P \mapsto [n]P = \underbrace{P \oplus P \oplus \cdots \oplus P}_{n \text{ fois}}$$

Cette définition est prolongée trivialement à tout $n \in \mathbb{Z}$, en prenant $[0]P = P_\infty$ et $[n]P = [-n](-P)$ pour $n < 0$.

1.1.5 Points rationnels sur une courbe elliptique

Soit E une courbe elliptique définie sur K. Les points de E à coordonnées dans K forment l'ensemble des points $K - rationnels$ de E notés $E(K)$. Nous avons

$$E(K) = \{(x_1, y_1) \in K^2 | y_1^2 + a_1x_1y_1 + a_3y_1 = x_1^3 + a_2x_1^2 + a_4x_1 + a_6\} \cup \{P_\infty\}.$$

1.1.6 Points de torsion

Définition 1.1.5. .
Soient E/K une courbe elliptique, $n \in \mathbb{Z}$. Le noyau de $[n]$, noté par $E[n]$, satisfait

$$E[n] = \{P \in E(K) | [n]P = P_\infty\}.$$

Un élément $P \in E[n]$ est appelé point $n - torsion$.

Théorème 1.1.1. .
Soit E une courbe elliptique définie sur K. Si la caractéristique de K est nulle ou premier à n alors

$$E[n] \simeq \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}.$$

Sinon, lorsque $char(K) = p$ et $n = p^r$, alors

$$E[p^r] = \{P_\infty\}, \text{ pour tout } r > 1 \text{ ou } E[p^r] \simeq \mathbb{Z}/p^r\mathbb{Z}, \text{ pour tout } r > 1.$$

1.1.7 Isogénies

Définition 1.1.6. .
Deux courbes E/K et E'/K sont isogènes sur K s'il existe un morphisme $\psi : E \to E'$ (avec les coefficients dans K) envoyant l'élément neutre de E sur l'élément neutre de E'. De cette propriété simple, il est possible de montrer que ψ est un homomorphisme de groupe de $E(K)$ dans $E'(K)$.

Proposition 1.1.1. .
Soit $\mathbb{F}_q$, un corps fini à q éléments.

Deux courbes elliptiques E et E' définies sur $\mathbb{F}_q$ sont isogènes sur $\mathbb{F}_q$ si et seulement si $|E(\mathbb{F}_q)| = |E'(\mathbb{F}_q)|$ (même cardinal).

1.1.8 Endomorphismes

La multiplication par n est un endomorphisme de la courbe E pour tout $n \in \mathbb{Z}$. L'ensemble de tous les endomorphismes de E définis sur K sera noté par $End_K(E)$ ou plus simplement par $End(E)$.

Soit $\mathbb{F}_q$, un corps fini à q éléments, $q = p^r$ avec p premier. Soit E une courbe elliptique sur $\mathbb{F}_q$. L'automorphisme de Frobenius de $\mathbb{F}_q$ s'étend à tous les points de la courbe en renvoyant P_∞ à lui-même et $P = (x_1, y_1)$ à $\phi_q(P) = (x_1^q, y_1^q)$. On peut facilement vérifier que le point $\phi_q(P)$ est aussi un point sur la courbe sans tenir compte du corps de définition de P. D'où, ϕ_q est un endomorphisme de E, appelé endomorphisme de Frobenius de $E/\mathbb{F}_q$. Il est différent de $[n]$ pour tout $n \in \mathbb{Z}$.

1.1.9 Cardinalité

Soit $\mathbb{F}_q$, un corps fini à q éléments. La cardinalité d'une courbe elliptique E sur $\mathbb{F}_q$, i.e., le nombre de points $\mathbb{F}_q - rationnels$, est un aspect important pour la sécurité des cryptosystèmes construits sur $E(\mathbb{F}_q)$.

Théorème 1.1.2. ***(Hasse, Weil)***.
Soit E une courbe elliptique définie sur $\mathbb{F}_q$. Alors, il existe t tel que

$$|E(\mathbb{F}_q)| = q + 1 - t \quad avec \quad |t| \leq 2\sqrt{q}.$$

Définition 1.1.7. ***(trace d'un endomorphisme)***.
La trace d'un endomorphisme est la trace de la matrice associée à cet endomorphisme dans une base quelconque.

Remarque 1.1.1. .

(i) L'entier t est la trace *de l'endomorphisme de Frobenius.*
(ii) Pour tout entier $t \in [-2\sqrt{p}, 2\sqrt{p}]$, il existe au moins une courbe elliptique E définie sur $\mathbb{F}_p$ de cardinalité $p + 1 - t$.

Théorème 1.1.3. .
Soit $q = p^r$. Il existe une courbe elliptique E définie sur $\mathbb{F}_q$ avec $|E(\mathbb{F}_q)| = q + 1 - t$ si et seulement si une des conditions suivantes est vérifiée :

1. *$t \not\equiv 0 \mod p$ et $t^2 \leq 4q$.*
2. *r est impair soit (i) $t = 0$ ou (ii) $p = 2$ et $t^2 = 2q$ ou (iii) $p = 3$ et $t^2 = 3q$.*
3. *r est pair et soit (i) $t^2 = 4q$ ou (ii) $p \not\equiv 1 (\mod 3)$ et $t^2 = q$ ou (iii) $p \not\equiv 1 (\mod 4)$ et $t = 0$.*

1.2 Courbes elliptiques de Huff non binaire

Il est connu que les courbes elliptiques peuvent être représentées sous différentes formes. Ces différentes formes induisent différentes propriétés arithmétiques. Pour obtenir des multiplications scalaires rapides, diverses formes de courbes elliptiques ont été étudiées dans les dernières décennies.

Il y a plusieurs moyens de représenter une courbe elliptique tels que :

1. Short Weierstrass : $y^2 = x^3 + ax + b$
2. Twisted Edwards : $ax^2 + y^2 = c^2[1 + dx^2y^2]$
3. Doche-Icart-Kohel : $y^2 = x^3 + 3a(x + 1)^2$
4. Jacobi intersection : $x^2 + y^2 = 1,\ ax^2 + z^2 = 1$
5. Jacobi quartic : $y^2 = x^4 + 2ax^2 + 1$
6. Legendre : $y^2 = x(x - 1)(x - \lambda)$
7. Montgomery : $by^2 = x^3 + ax^2 + x$
8. Hessian : $x^3 + y^3 + 1 = 3dxy$

Notons que certaines de ces courbes ont des points singuliers sur la clôture projective mais elles ont toutes un genre égal à 1.

Récemment, Joye, Tibouchi et Vergnaud ont revisité pour les corps finis, un modèle pour les courbes elliptiques sur $\mathbb{Q}$ introduit par Huff en 1948 dans le but d'étudier des problèmes diophantiens. Maintenant, la liste des formes de courbes elliptiques de Huff est la suivante :

1. Huff en 1948 : K un corps, $charc(K) \neq 2$, courbe : $ax(y^2-1) = by(x^2-1)$ avec $a^2 - b^2 \neq 0$, et $a,\ b \in K$

2. Joye, Tibouchi et Vergnaud en 2010 : K un corps, $charc(K) \neq 2$, Courbe : $ax(y^2-d) = by(x^2-d)$ avec $abd(a^2-b^2) \neq 0$ et $a,\ b,\ d \in K$;

3. Wu et Feng en 2011 : K un corps, $charc(K) \neq 2$, Courbe : $x(ay^2-1) = y(bx^2-1)$ avec $ab(a-b) \neq 0$, et $a,\ b \in K$;

4. Ciss et Sow en 2011 : K un corps, $charc(K) \neq 2$, Courbe : $ax(y^2-c) = by(x^2-c)$ avec $abcd(a^2c - b^2d) \neq 0$, et $a,\ b,\ c,\ d \in K$;

5. Joye, Tibouchi et Vergnaud 2010 : K un corps, $charc(K) \neq 2$, Courbe : $ax(y^2+y+1) = by(x^2+x+1)$ avec $abcd(a^2c - b^2d) \neq 0$;

6. Devigne et Joye en 2011 : K un corps, $charc(K) = 2$, Courbe : $ax(y^2+fy+1) = by(x^2+fx+1)$ avec $abf(a-b) \neq 0$.

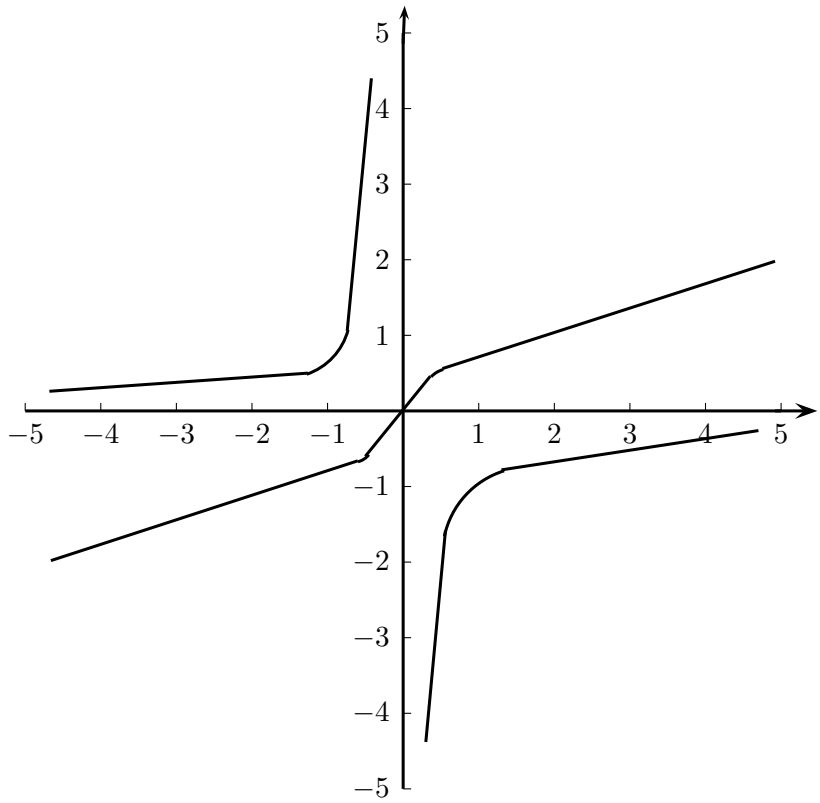
Définition 1.2.1. *Soit $\mathbb{K}$ un corps de characteristique $\neq 2$. Considérons l'ensemble des points projectifs $(X : Y : Z) \in \mathbb{P}^2(\mathbb{K})$ satisfaisant l'équation*

$$\mathcal{H}_{(a,b)/\mathbb{K}} : aX(Y^2 - Z^2) = bY(X^2 - Z^2)$$

où $a, b \in \mathbb{K}^\times$ et $a^2 \neq b^2$. Cette forme est référencée comme le modèle de Huff d'une courbe elliptique non binaire.

$\mathcal{O} = (0 : 0 : 1)$ est l'élément neutre de la loi de groupe et l'inverse d'un point $P = (X : Y : Z)$ est $-P = (X : Y : -Z)$.

Exemple d'un schéma d'une courbe de Huff sur $\mathbb{R}$: $ax(y^2-1) = by(x^2-1)$



1.2.1 Formules de l'équation affine

Nous donnons une formule explicite pour la loi de groupe. Nous utilisons la forme non homogène $ax(y^2-1) = by(x^2-1)$. Soit $y = \lambda x + \mu$ une droite affine passant par deux points distincts $P_1 = (x_1, y_1)$ et $P_2 = (x_2, y_2)$. Cette droite coupe la courbe en un troisième point $P_3 = (x_3, y_3)$. En remplaçant cette droite dans l'équation de la courbe, nous obtenons

$$\begin{cases} x_3 = x_1 + x_2 + \dfrac{\mu(2a\lambda - b)}{\lambda(a\lambda - b)} \\ y_3 = \lambda x_3 - \mu \end{cases}$$

avec $\lambda = \frac{y_1 - y_2}{x_1 - x_2}$ et $\mu = y_1 - \lambda x_1$.

On obtient la formule suivante après plusieurs calculs longs et fastidieux :

$$\begin{cases} x_3 = \dfrac{(x_1 + x_2)(1 + y_1 y_2)}{(1 + x_1 x_2)(1 - y_1 y_2)} \\ y_3 = \dfrac{(y_1 + y_2)(1 + x_1 x_2)}{(1 - x_1 x_2)(1 + y_1 y_2)} \end{cases}$$

avec $x_1 x_2 \neq \pm 1$ et $y_1 y_2 \neq \pm 1$.

Ainsi la formule d'addition peut aussi être utilisée pour calculer le point $2P = (x_3, y_3)$ étant donné $P = (x_1, y_1)$. On a ainsi,

$$\begin{cases} x_3 = \dfrac{2x_1(1 + y_1^2)}{(1 + x_1^2)(1 - y_1^2)} \\ y_3 = \dfrac{2y_1(1 + x_1^2)}{(1 - x_1^2)(1 + y_1^2)} \end{cases}$$

1.2.2 Formules de l'équation projective

Considérons maintenant P_1 et P_2 en coordonnées projectives, c'est-à-dire $P_1 = (X_1, Y_1, Z_1)$ et $P_2 = (X_2, Y_2, Z_2)$, et $\mathcal{O} = (0, 0, 1)$ comme élément neutre de la loi d'addition. Soit $P_3 = P_1 + P_2 = (X_3, Y_3, Z_3)$. Alors,

$$\begin{cases} X_3 = (X_1Z_2 + X_2Z_1)(Y_1Y_2 + Z_1Z_2)^2(Z_1Z_2 - X_1X_2) \\ Y_3 = (Y_1Z_2 + Y_2Z_1)(X_1X_2 + Z_1Z_2)^2(Z_1Z_2 - Y_1Y_2) \\ Z_3 = (Z_1^2Z_2^2 - X_1^2X_2^2)(Z_1^2Z_2^2 - Y_1^2Y_2^2) \end{cases} \qquad (*)$$

1.2.3 Applicabilité

Si $(x_1, y_1) \neq (0, 0)$ alors $(x_1, y_1) \oplus (a : b : 0) = -(\frac{1}{x_1}, \frac{1}{y_1})$ (car $(x_1, y_1) \simeq (x_1 : y_1 : 1)$). Notez que l'équation précédente reste valide pour le doublement du point $(a : b : 0)$ ou pour l'addition du point $(a : b : 0)$ à un autre point fini (i.e., qui n'est pas à l'infini) différent de $\mathcal{O}$, on obtient $(X_1 : Y_1 : Z_1) \oplus (a : b : 0) = (-Y_1Z_1 : -X_1Z_1 : X_1Y_1)$ comme prévu. La formule d'addition est toutefois invalide pour l'addition de $(0 : 1 : 0)$ ou $(1 : 0 : 0)$. Plus généralement, on obtient ce qui suit.

Théorème 1.2.1. *Soit $\mathbb{K}$ un corps de charactéristique $\neq 2$. Soient $P_1 = (X_1 : Y_1 : Z_1)$ et $P_2 = (X_2 : Y_2 : Z_2)$ deux points sur une courbe de Huff dans $\mathbb{K}$. Alors la formule d'addition donnée par les égalités ci-dessus $(*)$ est valide pourvu que $X_1X_2 \neq \pm Z_1Z_2$ et $Y_1Y_2 \neq \pm Z_1Z_2$.*

Corollaire 1.2.1. *Soit $\mathcal{H}$ une courbe de Huff sur un corps $\mathbb{K}$ de charactéristique impair. Soit $P \in \mathcal{H}(\mathbb{K})$ un point d'ordre impair. Alors la loi d'addition dans le sous-groupe généré par P est complète.*

1.2.4 Universalité du modèle

Le théorème suivant énonce que toute courbe elliptique sur un corps de charactéristique $\neq 2$ contenant une copie de $\mathbb{Z}/4\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$ peut être identifiée à une forme de Huff.

Théorème 1.2.2. *Toute courbe elliptique $(E, \mathcal{O})$ sur un corps parfait $\mathbb{K}$ de charactéristique $\neq 2$ tel que $E(\mathbb{K})$ contient un sous-groupe G isomorphe à $\mathbb{Z}/4\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$, est équivalente birationnellement sur $\mathbb{K}$ à une courbe de Huff.*

1.3 Courbes elliptiques de Huff binaire

Dans le but d'étudier les problème diophantiens, Huff a introduit un nouveau modèle de courbe elliptique [22] (voir aussi [35]). Récemment, le modèle de Huff a été revisité dans [24]. Dans cette section, nous spécifions la loi de groupe sur les courbes de Huff binaire introduites par J. Devigne et M. Joye [10]. Nous allons commencer par une définition.

Définition 1.3.1. *([24]). Une courbe de Huff binaire est l'ensemble des points projectifs* $(X : Y : Z) \in P_2(\mathbb{F}_{2^m})$ *satisfaisant l'équation*

$$E/\mathbb{F}_{2^m} : aX(Y^2 + YZ + Z^2) = bY(X^2 + XZ + Z^2) \tag{1}$$

où $ab \in \mathbb{F}_{2^m}$ *et* $a \neq b$.

Il y a trois points à l'infini satisfaisant l'équation de la courbe, c'est à dire$(a : b : 0), (1 : 0 : 0)$, *et* $(0 : 1 : 0)$. *Le modèle affine correspondant à la courbe de Huff binaire donnée par l'équation* (1) *est :*

$$ax(y^2 + y + 1) = by(x^2 + x + 1).$$

Cette courbe est birationnellement équivalente à une courbe elliptique de Weierstrass de la forme : $v(v + (a + b)u) = u(u + a^2)(u + b^2)$ *via les fonctions inverses l'une de l'autre.*

$$(x, y) \leftarrow \left(\frac{b(u + a^2)}{v}, \frac{a(u + b^2)}{v + (a + b)u}\right) \text{ et } (u, v) \leftarrow \left(\frac{ab}{xy}, \frac{ab(axy + b)}{x^2y}\right) \tag{2}$$

L'ensemble des points de la courbe de Huff binaire forme un groupe. L'élément neutre est $\mathcal{O}$.

1.3.1 Formule d'addition

Soit $P = (x_1, y_1)$ et $Q = (x_2, y_2) \in E$ deux points finis avec $P \neq Q$. Alors l'addition de P et Q est donné par $P + Q = (P * Q) * \mathcal{O}$, (avec $P * Q = R$ le troisième point d'intersection de la droite (PQ) et la courbe). Alors, on obtient $P + Q = (x_3, y_3)$ avec

$$x_3 = \frac{(x_1y_1 + x_2y_2)(1 + y_1y_2)}{(y_1 + y_2)(1 + x_1x_2y_1y_2)} \text{ et } y_3 = \frac{(x_1y_1 + x_2y_2)(1 + x_1x_2)}{(x_1 + x_2)(1 + x_1x_2y_1y_2)} \tag{5}$$

Si $x_1 = x_2$ alors $P + Q = (0 : 1 : 0)$. Si $y_1 = y_2$ alors $P + Q = (1 : 0 : 0)$.

Pour l'inverse et le doublement voir [10].

1.3.2 Universalité de la loi d'addition

La formule d'addition présentée précédemment pour évaluer $P + Q$ distingue deux cas : $P = Q$ (doublement) et $P \neq Q$ (addition). Dans cette partie, nous donnons la formule d'addition qui peut être utilisée dans les deux cas. L'opération correspondante est appelée addition unifiée.

La formule suivante est celle de l'addition unifiée.

Soit $P = (x_1, y_1)$ et $Q = (x_2, y_2) \in E$ deux points finis, avec $P \neq Q$. L'équation (5) énonce la somme de deux points $P + Q = (x_3, y_3)$ où

$$x_3 = \frac{(x_1y_1 + x_2y_2)(1 + y_1y_2)}{(y_1 + y_2)(1 + x_1x_2y_1y_2)} \text{ et } y_3 = \frac{(x_1y_1 + x_2y_2)(1 + x_1x_2)}{(x_1 + x_2)(1 + x_1x_2y_1y_2)}$$

L'addition de deux points est déterminée via l'équation de la courbe. En utilisant les relations d'addition $ax_i(y_i^2 + y_i + 1) = by_i(x_i^2 + x_i + 1)$, $i \in \{1, 2\}$, nous avons :

$$\begin{cases} x_3 = \dfrac{b(x_1+x_2)(1+x_1x_2y_1y_2)+(a+b)x_1x_2(1+y_1y_2)}{b(1+x_1x_2)(1+x_1x_2y_1y_2)} \\ y_3 = \dfrac{a(y_1+y_2)(1+x_1x_2y_1y_2)+(a+b)y_1y_2(1+x_1x_2)}{a(1+y_1y_2)(1+x_1x_2y_1y_2)} \end{cases} \tag{6}$$

Remarquez que cette nouvelle expression fonctionne aussi pour le doublement d'un point $P = (x_1, y_1)$.

En effet, si on remplace (x_2, y_2) avec (x_1, y_1) dans l'équation (6), on obtient :

$$x_3 = \frac{(a+b)x_1^2(1+y_1^2)}{b(1+x_1^2)(1+x_1^2y_1^2)} \text{ et } y_3 = \frac{(a+b)y_1^2(1+x_1^2)}{a(1+y_1^2)(1+x_1^2y_1^2)}$$

c'est-à-dire la formule du doublement précédente de l'équation (4).

Chapitre 2

Cryptographie et Preuves de Sécurité

2.1 Complexité

2.1.1 Notions de complexité

L'objet de la théorie de la complexité est :

- pour les algorithmes d'évaluer le nombre d'opérations élémentaires (complexité temporelle) et l'espace mémoire nécessaire (complexité spatiale) pour leur résolution ;
- pour les problèmes (de décision), de les classifier suivants leur niveau de difficulté.

La principale préoccupation, c'est le temps de calcul en fonction de la taille des données d'entrée dans les algorithmes car le problème de l'espace mémoire se pose de moins en moins avec le développement de la technologie.

Classes de complexité

Définition 2.1.1. *Soient f et g deux fonctions à variable entière et à valeurs positives.*

$f(n) = O(g(n))$ s'il existe une constante $c > 0$ et un entier n_0 tels que $\forall n \geq n_0$, $0 \leq f(n) \leq cg(n)$. On dit que g domine f ou que f ne croit pas plus vite que g multiplié par une constante.

Types de complexité : On note d les données de taille $\leq n$, $C(d)$ le coût associé à l'exécution de la donnée d par un algorithme et $C(n)$ la complexité de l'algorithme. alors on peut définir $C(n)$ de trois façons différentes ;

1. **Complexité dans le pire des cas** : $C(n) = Max\{C(d), d = \text{ donnée de taille } \leq n\}$
2. **Compléxité en moyenne** : $C(n) = \sum_{d=\text{donnée de taille} \leq n} P(d)C(d)$ où $P(d)$ est la probabilité d'obtenir la donnée d
3. **Compléxité dans le meilleur des cas** : $C(n) = Min\{C(d), d = \text{donnée de taille} \leq n\}$

Pour évaluer la complexité ici on considère celle dans le **pire des cas** car on est seulement intéressé par les ordres de grandeur.

Définition 2.1.2. .

1. *Un algorithme est (à **complexité) polynômiale** si sa complexité dans le pire des cas est de la forme $O(n^k)$ où k est une constante.*
2. *Si sa complexité ne peut être majorée par un polynôme, on dit que l'algorithme est (à **complexité) exponentielle**.*
3. *Un algorithme est (à complexité) quasi (ou **sous) - exponentielle** si sa complexité est une fonction de la forme $e^{O(n)}$.*

   ***Exemple :** La fonction $L_q[\alpha, c] = O\big(exp\big((c+o(1))(\ln q)^{\alpha}(\ln\ln q)^{1-\alpha}\big)\big)$ où $c > 0$ et $0 < \alpha < 1$ mesure la complexité sous-exponentielle. car sa complexité est :*
   - *polynomiale en $\ln q$ pour $\alpha = 0$: $L_q[0, c] = O\big((\ln q)^{c+O(1)}\big)$*
   - *exponentielle en $\ln q$ (car polynomiale en q) pour $\alpha = 1$: $L_q[1, c] = O\big(q^{c+O(1)}\big)$*

Classification et problèmes de décision

Problèmes de décision :

En théorie de la complexité, on s'intéresse aux problèmes dits de décision.

Problèmes de décision et classe de complexité :

- Un problème de décision est un problème dont la réponse est OUI ou NON.
- **La classe de complexité P** est l'ensemble des problèmes de décision résoluble en temps polynômiale.
- **La classe de complexité NP** est l'ensemble des problèmes de décision pour lesquels la réponse OUI peut être vérifiée en temps polynômiale moyennant une information supplémentaire appelée "certificat".
- **La classe de complexité $\mathbf{Co-NP}$** est l'ensemble des problèmes de décision pour lesquels la réponse NON peut être vérifiée en temps polynômial moyennant un "certificat" approprié.

Remarque 2.1.1. *Pour le problème du certificat, la définition* **NP** *(resp :* $\mathbf{Co-NP}$*) ne stipule pas que le certificat peut être trouvé en temps polynômial mais que s'il existe et s'il est connu, alors il peut être utilisé en temps polynômial pour vérifier qu'on a OUI (resp : NON).*

Réduction :

Soient $\mathbb{D}_1$ et $\mathbb{D}_2$ deux problèmes de décisions. On dit que $\mathbb{D}_1$ est ***polynomialement réductible*** à $\mathbb{D}_2$, (noté : $\mathbb{D}_1 \preceq_{\mathbf{P}} \mathbb{D}_2$) s'il existe un algorithme $\mathcal{A}_1$ utilisant éventuellement un algorithme $\mathcal{A}_2$ qui résout $\mathbb{D}_2$ et vérifiant les deux conditions suivantes :

- le nombre d'appels de $\mathcal{A}_1$ à $\mathcal{A}_2$ est majoré par une fonction polynomiale de la donnée d'entrée ;
- le coût de $\mathcal{A}_1$ (hors appels à $\mathcal{A}_2$) est polynomial ;

Ainsi, si $\mathcal{A}_2$ résout $\mathbb{D}_2$ en temps polynômial alors $\mathcal{A}_1$ résout $\mathbb{D}_1$ en temps polynômial.

On interprète, $\mathbb{D}_1 \preceq_{\mathbf{P}} \mathbb{D}_2$ en disant que $\mathbb{D}_2$ est aussi difficile que $\mathbb{D}_1$ où que $\mathbb{D}_1$ n'est pas plus dur que $\mathbb{D}_2$.

Equivalence :

On dit que deux problèmes $\mathbb{D}_1$ et $\mathbb{D}_2$ de décision sont *polynômialement équivalents* s'ils sont (mutuellement) polynômialement réductibles l'un à l'autre c'est à dire : $\mathbb{D}_1 \preceq_{\mathbf{P}} \mathbb{D}_2$ et $\mathbb{D}_2 \preceq_{\mathbf{P}} \mathbb{D}_1$

Problème NP-complet :

Un problème de décision est dit **NP**-complet si :

i) $\mathbb{D} \in \mathbf{NP}$

ii) $\mathbb{D}_1 \preceq_{\mathbf{P}} \mathbb{D}$ pour tout $\mathbb{D}_1 \in \mathbf{NP}$.

Les problèmes **NP**-complets sont les problèmes les plus difficiles dans **NP** c'est à dire qu'ils sont aussi difficiles que tout autre problème de **NP**

Problème NP-Dur :

Un problème $\mathbb{D}$ est **NP-Dur** s'il existe un problème $\mathbb{H}$ qui est **NP-Complet** et qui se réduit *polynômialement* à $\mathbb{D}$ c'est à dire $\mathbb{H} \preceq_{\mathbf{P}} \mathbb{D}$.

Un problème **NP-Dur** n'est pas forcément un problème de décision mais il est plus difficile qu'un problème **NP-Complet**. D'ailleurs tout problème **NP-Complet** est **NP-Dur**.

2.1.2 Fonctions à sens unique, fonctions à sens unique avec trappe

- Une fonction $f : A \longrightarrow B$ est dite à sens unique s'il est facile de calculer $f(x)$, $\forall\ x \in A$ (complexité polynômiale) et il est difficile (complexité exponentielle) étant donné y de trouver x tel que $y = f(x)$.
- Une fonction est à sens unique est avec trappe si l'on connaît un secret permettant de l'inverser.

Exemples 2.1.1. *Soit G un groupe cyclique et q sa taille et g un générateur de G.*

$$f : [0, q-1] \longrightarrow G$$

$$x \longmapsto g^x$$

f est à sens unique si G est un groupe cyclique, de cardinal assez grand de sorte que connaissant y, la résolution de $y = g^x$ est difficile pour x secret : c'est le problème du logarithme discret (DLP).

NB : il existe plusieurs algorithmes pour résoudre le logarithme discret mais ils sont tous exponentiels ou quasi-exponentiels en la taille q de G.

Exemples 2.1.2.

$$f : (\mathbb{Z}/n\mathbb{Z})^* \longrightarrow (\mathbb{Z}/n\mathbb{Z})^*$$

$$x \longmapsto x^e$$

f est à sens unique avec trappe si $n = pq$ produit de deux nombres premiers très grands de même taille de sorte que la factorisation soit difficile et $e \wedge \varphi(n) = 1$ où $\varphi(n) = (p-1)(q-1)$ avec p, q et $\varphi(n)$ secrets.

Ici, la trappe correspond à la connaissance de $\varphi(n)$ car on peut calculer d tel que $ed = 1 \mod \varphi(n)$ avec l'algorithme étendu d'Euclide. Dans ce cas $y = x^e \mod n$ si et seulement si $x = y^d \mod n$.

Il existe plusieurs algorithmes pour factoriser mais ils sont tous exponentiels ou sous-exponentiels.

NB : La résolution de $y = x^e \mod n$ est appelé problème RSA et paraît difficile tant que la factorisation reste difficile. On ne sait pas si les deux problèmes sont équivalents.

2.2 Généralités sur la cryptographie

La cryptographie est composée de la cryptographie à clé secrète et de la cryptographie à clé publique. La cryptographie à clé publique est inventée en 1976 par Diffie et Hellman dans un article aujourd'hui légendaire [11].

D'autres font remonter l'invention de la cryptographie à clé publique à une date plus ancienne.

Elle se distingue de la cryptographie à clé secrète sur au moins trois points.

- en cryptographie à clé secrète les deux parties en communication partagent le même secret alors qu'en cryptographie à clé publique chaque partie a son propre secret appelé clé privée et une information associée accessible à tout le monde appelée clé publique ;

- la cryptographie à clé secrète utilise essentiellement les fonctions booléennes ($f : \{0,1\}^n \longrightarrow \{0,1\}$) et les statistiques alors que la cryptographie à clé publique s'appuie beaucoup plus sur l'arithmétique et la théorie algébrique des nombres ;

- la cryptographie à clé secrète est utilisée essentiellement pour la confidentialité alors que la cryptographie à clé publique peut en plus garantir la signature numérique et d'autres services.

Les performances de la cryptographie à clé publique sont sensiblement inférieures à celles de la cryptographie à clé secrète : c'est peut-être le prix à payer pour de nouveaux services de sécurité.

2.2.1 Système de cryptographie

Un système de cryptographie est composé d'un quintuplet $(\mathcal{P}, \mathcal{E}, \mathcal{E}_k, D_{k'}, \mathcal{K})$ où :

- $\mathcal{P}$ est un ensemble appelé espace des textes clairs

- $\mathcal{E}$ est un ensemble appelé espace des textes chiffrés

- $\mathcal{K}$ est un ensemble appelé espace des clés

- $Gen_{\mathcal{K}}$ un algorithme de génération de clés (=les éléments de $\mathcal{K}$) ;

- $\mathcal{E}_k : \mathcal{P} \to \mathcal{E}$ est une fonction inversible à gauche appelée fonction de chiffrement et qui dépend d'un paramètre k appelé clé.

- $D_{k'} : \mathcal{E} \to \mathcal{P}$ est la fonction inverse à gauche de $\mathcal{E}_k$ (i.e $D_{k'} \circ \mathcal{E}_k(m) = m, \forall m \in \mathcal{P}$) et est appelée fonction de déchiffrement (dépendant de la clé k'.)

A partir de ce modèle on a deux cryptosystèmes.

2.2.2 Système de cryptographie à clé secrète

Le système à clé secrète correspond au cas suivant :

- $k = k'$ ou chaque clé est calculatoirement facile à déduire de l'autre ;
- Ainsi il est nécessaire de garder secret les clés et d'avoir un canal (une méthode) sûr de communication pour partager la clé secrète.

Les algorithmes de chiffrements symétriques sont utilisés pour rendre essentiellement le service de confidentialité et sont composés de deux catégories : les algorithmes de chiffrements par blocs (DES [13], AES [12], $\cdots$) et les algorithmes de chiffrements par flux (RC4, Trivium [9], $\cdots$).

2.2.3 Système de cryptographie à clé publique

Le système à clé publique correspond au cas suivant :

- $k \neq k'$ et l'une des clés (soit k') est difficile à calculer à partir de k ;

- k est alors publiée et est appelée clé publique, k' est gardée secrète par le propriétaire et est appelée clé privée.

- Il n'y a pas de nécessité de canal sûr pour échanger les clés mais, il faut que les utilisateurs puissent s'assurer de l'authenticité des clés publiques.

- Ainsi, il est nécessaire d'avoir un tiers de confiance (autorité de certification capable de certifier la validité d'une clé publique d'une entité bien identifiée) ou une chaîne de confiance comme dans le cas de l'utilisation de GnuPG.

Comme exemples d'algorithme à clé publique, on a : RSA, Mc-Eliece [31], El Gamal etc

Nous présentons ici les deux cryptosystèmes asymétriques les plus classiques : RSA [41] et El Gamal [17]. Ces deux systèmes ont l'avantage d'être très simples à décrire, du moins dans leur version mathématique.

Chiffrement RSA

Du nom de ses inventeurs (Rivest, Shamir et Adleman), le cryptosystème RSA [41] s'appuie sur la difficulté de la factorisation.

Algorithme de génération des clés

- On choisit un entier n(modulo) tel que $n = pq$ avec p et q deux nombres premiers assez grands ($|p| = |q| \geq 1024\ bits$) ;
- On calcule $\varphi(n) = (p-1)(q-1)$. φ est l'indicateur d'Euler et $\varphi(n)$ le nombre d'éléments inversibles de $\frac{\mathbb{Z}}{n\mathbb{Z}}$ (c'est à dire le cardinal du groupe $(\frac{\mathbb{Z}}{n\mathbb{Z}})^*$) ;
- On choisit e tel qu'il soit plus petit que $\varphi(n)$ et $\text{pgcd}(e, \varphi(n)) = 1$;
- On calcule d tel que $ed = 1 \mod \varphi(n)$ (avec l'algorithme étendu d'Euclide).
- Clé publique (e, n) : (généralement $e = 2^{16} + 1$ est fixé) ;
- Clé privée (d, n). p, q et $\varphi(n)$ doivent rester secrets (on peut détruire $\varphi(n)$) .

Algorithme de chiffrement

– Soit $M \in \frac{\mathbb{Z}}{n\mathbb{Z}}$ un message, alors le chiffré est : $C = M^e \mod n$;

Algorithme de déchiffrement

– On déchiffre un message chiffré $C \in \frac{\mathbb{Z}}{n\mathbb{Z}}$ par : $C^d \mod n = M$, (si le déchiffrement est valide) ;

– on utilise l'algorithme de calcul de puissance rapide car d est très grand en général.

Chiffrement El Gamal

Nous présentons El Gamal dans un groupe G quelconque.

Algorithme de génération des clés

– Choisir G un groupe et $g \in G$ d'ordre premier q, $(n = \#G,\ |n| = 1024,\ |q| \geq 190,\ q\ divise\ n)$;

– Alice choisit a aléatoirement dans $]1, q-1[$, et calcule $h = g^a$ dans G, $(|a| = |q|)$;

– La clé publique est (g, h, G) ;

– La clé secrète est a.

Algorithme de chiffrement

– Clé publique est (g, h, G) ;

– Choisir un nombre aléatoire $k \in]1, q-1[$ assez grand, $(|k| = |q|)$;

– Pour chiffrer un message clair $m \in \langle g \rangle$, Bob calcule :

  – $m_1 = g^k \in G$;

  – $m_2 = m.h^k \in G$.

– Chiffré $c = (m_1, m_2)$.

Algorithme de déchiffrement

– Clé privée a ;

– Alice reçoit le chiffré $c = (m_1, m_2)$;

– Elle calcule : $m' = m_1^{n-a} \cdot m_2 \in G$.

2.2.4 Fonctions de hachage

Définitions :

Une fonction de hachage est une fonction publique $h : \{0,1\}^* \to \{0,1\}^n$ telle que :

a) h transforme un message (binaire) de longueur quelconque en un message de longueur fixe, (Fonction de Compression) ;

b) pour tout x, $h(x)$ est facile à calculer, (Facilement calculable) ;

Les images $h(x)$ sont appelés *hache* ou *empreinte*.

On dira que c'est une fonction de hachage pour la cryptographie si en plus la fonction a les propriétés suivantes :

1. pour presque tout y, il est difficile de trouver x tel que $h(x) = y$, (Fonction à sens unique sans trappe) ;

2. Pour presque tout x, il est difficile de trouver x' tel que $h(x) = h(x')$, (Résistante aux collusions) ;
3. Il est difficile de trouver un couple (x, x') tel que $h(x) = h(x')$, (Fortement résistante aux collusions).

Intégrité (MAC) :

Si x est un message alors pour garantir l'intégrité de x on envoie ou on stocke le couple $(x, h(x, k))$ où $h(x, k)$ est l'empreinte de x via une fonction de hachage h utilisant une clé k. Le message est considéré intègre s'il est bien accompagné par son empreinte qu'on ne peut falsifier sans connaitre k (connaissant x et $h(x, k)$).

Classification

Les fonctions de hachage comptent deux familles

- celles utilisant des clés ;

- celles n'utilisant pas de clés.

Usage

Les fonctions de hachage sont utilisés pour :

– l'intégrité
– construire des générateurs aléatoires cryptographiquement sûr ;
– pour la modélisation théorique des fonctions à sens unique telles que le modèle de l'oracle aléatoire dans les preuves de sécurité.

Principaux algorithmes de hachage

Les principaux algorithmes de hachage sont : SHA1, SHA2, SHA3 [15].

2.2.5 Système de signature

Une signature est un procédé, qui, appliqué à un message, garantit la non répudiation par le signataire et donc réalise les deux objectifs suivants :

– identification unique du signataire,
– et preuve d'accord sur le contenu du document.

Elle doit posséder les propriétés suivantes :

1. unique
2. impossible à usurper
3. impossible à répudier par son auteur,
4. facile à vérifier par un tiers,
5. facile à générer

Un système de signature cryptographique est composé d'un 6-tuplet $(\mathcal{P}, \mathcal{H}, \mathcal{S}, S_{k'}, V_k, \mathcal{K})$ où :

- $\mathcal{P}$ est un ensemble appelé espace des textes à signer ;

- $\mathcal{S}$ est un ensemble appelé espace des signatures ;

- $h : \mathcal{P} \longrightarrow \mathcal{H}$ une fonction de hachage ;

- $\mathcal{K}$ est l'espace des clés ;

- $S_{k'} : \mathcal{H} \longrightarrow \mathcal{S}$ est une fonction injective dite fonction de signature (non nécessairement bijective) qui dépend de la clé privée k' ;

- $V_k : \mathcal{P} \times \mathcal{S} \rightarrow \{vraie, faux\}$ est la fonction de vérification de signature binaire telle que $V_k(m, s) =$ vraie si et seulement si $S_{k'}(h(m)) = s$ (la vérification dépendant de la clé publique k).

Nous donnons ici deux exemples :

Signature RSA avec RSA-FDH

Signature : le cryptosystème RSA-FDH (Full Domain Hash) s'utilise aussi pour générer des signatures électroniques. Une signature électronique garantit l'authenticité, l'intégrité et la non-répudiation d'un document électronique. Pour signer un message $m \in \mathbb{Z}_n$ avec RSA, le signataire utilise sa clé privée (d, n) pour obtenir la signature :

$S = h(m)^d \mod n$

Vérification Seul le signataire possédant la clé privée (d, n) est donc capable de signer le message m. On vérifie la validité de la signature S en utilisant la clé publique (e, n), en s'assurant que :

$h(m) = S^e \mod n$

Signature El Gamal

Signature avec hachage

Soit G un groupe fini de grande taille.

Algorithme de génération de clés :

L'entité Alice

1. sélectionne un sous groupe cyclique H de G d'ordre premier q, avec un générateur g. (supposons que G est noté multiplicativement.)
2. sélectionne un entier aléatoire secret a, $1 \leq a \leq q-1$ et calcule $y = g^a$ dans G.
3. la clé publique est (g, y, G), et la clé privée est a.

Algorithme de signature et de vérification

1. Signature d'un message m : Alice
   (a) sélectionne un entier aléatoire secret k, $1 \leq k \leq q-1$, avec $pgcd(k, q) = 1$.
   (b) calcule l'élément de groupe $r = g^k$.
   (c) calcule $k^{-1} \mod q$ (en utilisant l'algorithme d'Euclide étendu).
   (d) calcule $h(m)$ et $h(r)$.
   (e) calcule $s = k^{-1}\{h(m) - ah(r)\} \mod q$.

(f) la signature de m est le couple (r, s).

2. Vérification de la signature (r, s) du message m : Bob
   (a) prend la clé publique du signataire (g, y).
   (b) calcule $h(m)$ et $h(r)$.
   (c) calcule $v_1 = y^{h(r)} \cdot r^s$.
   (d) calcule $v_2 = g^{h(m)}$.
   (e) accepte la signature si et seulement si $v_1 = v_2$.

2.3 Cryptographie à clé publique et preuve de sécurité

Pour de plus amples informations sur les preuves de sécurité, voir le cours de D. Pointcheval [37, 38] et D. Vergnaud [45].

2.3.1 Notions de base

La preuve de sécurité est une démarche mathématique qui permet de démontrer qu'un système est sûr par rapport à un modèle d'attaquant donné (moyens et puissance de calcul).

Les preuves de sécurité dépendent de la cryptographie en question :

1. pour la cryptographie symétrique : c'est la sécurité inconditionnelle (la puissance de calcul de l'attaquant est illimitée) ;
2. pour la cryptographie non symétrique (clé publique, fonction de hachage, signature) : c'est la sécurité réductionniste (on transforme toute réussite d'une attaque en la solution d'une instance d'un problème réputé difficile).
   a) clés publiques et signatures : $\longrightarrow$ problèmes difficiles $\longleftarrow$ théorie des nombres ;
   b) fonctions de hachage : $\longrightarrow$ fonctions aléatoires (fonction booléenne $f : \{0,1\}^* \longrightarrow \{0,1\}^n$).

Comme on travaille sur la cryptographie à clé publique, on va donner quelques rappels sur les preuves de sécurité.

Sécurité réductionniste

- Temps de calcul raisonnable : temps de calcul effectué par un algorithme polynômial déterministe ou probabiliste.
- Algorithme polynômial : algorithme dont le temps de calcul est majoré par une fonction polynômiale en la taille des données d'entrée.
- Problèmes difficiles : problèmes pour lesquels il existe une large classe d'instances telle que aucun algorithme connu ne peut résoudre une de ces instances en un temps polynômial.

On prend en compte deux aspects :

- ce qu'on exige être impossible à calculer en temps raisonnable ;
- ce qu'on suppose à la disposition de l'attaquant pour tenter ce calcul.

"Si l'on suppose l'existence d'un algorithme A qui casse en pratique les instances de taille t d'un système S, on en déduit qu'on peut construire un algorithme B qui casse une instance de taille t' d'un problème P, réputé difficile".

Cette technique de réduction telle que présentée ici n'est intéressante que lorsque le temps de construire B sous l'hypothèse de l'existence de A est polynômial (=temps de réduction). Dans la pratique de la modélisation il y a deux approches :

- modèle standard : on réduit comme ci-dessus la sécurité du protocole ou de l'algorithme à celui de la difficulté d'un problème tel que la factorisation RSA, le logarithme discret, etc ... .
- modèle de l'oracle aléatoire : on suppose l'existence de fonction parfaitement aléatoire (qu'on implémente via des fonctions de hachage).

2.3.2 Modèle de l'oracle aléatoire

- **Oracle :** c'est un algorithme (machine turing ou une fonction f) auquel on peut soumettre une entrée x et recevoir une sortie $f(x)$ et que cet algorithme se comporte comme une boîte noire pour le requérant c'est à dire que le requérant n'exécute pas lui même f mais a simplement accès à tout $f(x)$ correspondant au x de son choix.

Généralement on peut supposer que :

- le requérant ne connaît pas une expression algébrique (ainsi que tous les paramètres utilisés) de la fonction comme dans le cas d'une fonction de déchiffrement. Ainsi quand un requérant est capable de faire déchiffrer le chiffré de son choix sans forcément savoir comment on déchiffre, on dit qu'il a accès à un oracle de déchiffrement ;

- le requérant connaît une expression algébrique de la fonction mais les sorties de cette fonction se comportent de façon aléatoire, on dit que c'est un oracle aléatoire (c'est le cas d'une fonction de hachage idéalisée).

- **Modélisations mathématiques :** on note $\{0,1\}^n$ l'ensemble des mots de longueur n sur $\{0,1\}$ et on note $\{0,1\}^*$ l'ensemble des mots de longueur finie : $\{0,1\}^* = \bigcup_{i\geq 0}\{0,1\}^n$ où $\{0,1\}^0$ est le mot vide. On note $\{0,1\}^\infty$ l'ensemble des suites infinies sur $\{0,1\}$.

On considère

$$\mathcal{P} = \{f : \{0,1\}^* \longrightarrow \{0,1\}^\infty\}$$

f prend en entrée un mot de longueur finie et produit une suite de longueur infinie.

- Une fonction $f \in \mathcal{P}$ est dite prise au hasard c'est-à-dire aléatoire si pour tout $x \in \{0,1\}^*$, chaque bit de la suite infinie $f(x)$ est prise au hasard c'est à dire $p(0) = p(1) = 1/2$.

- "Un oracle aléatoire est une procédure qui permet pour $x \in \{0,1\}^*$ de produire $f(x)$ où $f \in \mathcal{P}$ est une fonction aléatoire, sans révéler d'aucune façon le procédé de calcul."

Utilisation d'oracle aléatoire :

1) Dans les preuves de sécurité : Par exemple si un attaquant ne peut casser un chiffré même s'il a accès à un oracle de déchiffrement alors cela signifie qu'il n'a pas mieux à faire pour trouver

le message clair correspondant au chiffré qu'il veut attaquer (appelé challenge) que de choisir le message clair au hasard.

2) Construction d'une fonction de hachage : Si f est un oracle aléatoire, on peut définir une fonction de hachage $g : \{0,1\}^k \longrightarrow \{0,1\}^s$, en posant $g(x) =$ les s premiers bits de $f(x)$ pour $x \in \{0,1\}^k$;

3) Si $f : \{0,1\}^* \longrightarrow \{0,1\}^\infty$ est un oracle aléatoire et $\varphi : \{0,1\}^\infty \longrightarrow \{0,1\}^\infty$ est une bijection c'est à dire une renumérotation alors $\varphi \circ f$ est un nouveau oracle aléatoire.

2.3.3 Modèles d'adversaires

On va présenter les différents types d'attaques ainsi que les moyens associés.

Attaque à textes clairs choisis (chosen plaint text attack : CPA)

L'attaquant peut obtenir les chiffrés de son choix. Par exemple c'est un cas trivial en cryptographie à clé publique car la clé publique est accessible.

Attaque non adaptative à chiffrés choisis (nonadaptative chosen cipher text attack : CCA 1)

En plus de la clé publique, l'attaquant a accès à un oracle de déchiffrement uniquement avant de recevoir le challenge c'est à dire le chiffré à attaquer (donc il ne pourra pas adapter ses requêtes à l'oracle en fonction des informations qu'il a sur le challenge).

Attaque adaptative à chiffrés choisis (adaptative chosen cipher text attack : CCA 2)

En plus de la clé publique, l'attaquant peut accéder à un oracle de déchiffrement avant et après avoir reçu le challenge. Mais le challenge n'est pas soumis à l'oracle. Cette attaque est plus forte que les précédentes car après avoir pris connaissance du challenge il peut adapter les chiffrés qu'il souhaite faire déchiffrer.

2.3.4 Formalisme des preuves de sécurité des chiffrements à clé publique

Introduction

Il existe plusieurs modèles de preuves de sécurité :

- Indistinguabilité (IND) ;
- Sécurité sémantique ;
- Non malléabilité basée sur la comparaison (NM-C) ;
- Non malléabilité basée sur la simulation (NM-S).

Dans la suite, nous présenterons l'indistinguabilité.

Algorithme à clé publique comme fonction à sens unique (one way function)

Considérons un attaquant disposant d'un algorithme polynômial et probabiliste $\mathcal{A}$.

$\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ est un système à clé publique dont le paramètre de sécurité est noté k, $\mathcal{E}$ est l'algorithme de chiffrement et $\mathcal{D}$ l'algorithme de déchiffrement. $\mathcal{R}$ est l'algorithme qui résoud une instance d'un problème difficile $\mathcal{P}$ avec une probabilité $\varepsilon_{\mathcal{R}}$ en un temps $\tau_{\mathcal{R}}$ pour le paramètre de sécurité k.

On note $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ l'algorithme de l'attaquant.

Expérience : $Exp^{OW}(\mathcal{A}, k)$

$(p_k, s_k) \longleftarrow \mathcal{K}(1^k)$

$r \longleftarrow \mathcal{R}$

$c \overset{rand}{\longleftarrow} \mathcal{E}(p_k, m, r)$, (rand : random)

$r' \longleftarrow \mathcal{R}$

$\widehat{m} \longleftarrow \mathcal{A}(p_k, r', c)$

si $\widehat{m} = m$ retourner 1

si $\widehat{m} \neq m$ retourner 0

Fonction négligeable

Une fonction $f : \mathbb{N} \longrightarrow \mathbb{R}$ est négligeable si pour tout polynôme P à variable et valeur entières, il existe un entier $N(P)$ tel que $|f(x)| \leq \dfrac{1}{|P(n)|},\ \forall n \geq N(P)$.

Modélisation

L'attaquant est modélisé par un algorithme $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ composé de deux algorithmes : $\mathcal{A}_1$ s'exécute avant la réception du challenge c^* (à déchiffrer) et $\mathcal{A}_2$ après.

Chaque $\mathcal{A}_i$ peut faire appel à un oracle $\mathcal{O}_i$ avec un algorithme probabiliste polynômial. La sortie de l'algorithme $\mathcal{A}$ est celle de $\mathcal{A}_2$ et son entrée est celle de $\mathcal{A}_1$.

L'attaquant sera efficace si sa probabilité de succès est significativement plus forte que la probabilité de succès d'une réponse donnée avec le maximum de vraissemblance (sans connaître le challenge c^* et sans avoir accès à un oracle de déchiffrement).

Avantage de l'attaquant :

Soit MV un algorithme qui répond aléatoirement $|Pr(\mathcal{A}_2 = \text{ bonne réponse}) - Pr(MV = \text{ bonne réponse})|$ est l'avantage de l'attaquant.

En principe : $Pr(MV = \text{ bonne réponse}) = 1/2$ et l'avantage est :

$$Adv(\mathcal{A} = 2|Pr(\mathcal{A}_2 = \text{ bonne réponse}) - 1|$$

Sécurité sémantique

La sécurité sémantique stipule qu'un attaquant ne peut tirer aucune information du chiffré même s'il connaît un ensemble fini de textes clairs $\mathcal{M}$ (ces textes clairs sont les déchiffrés de chiffrés connus).

a) **Modélisation :**

Expérience :

Etape 1 :

Le challenger déroule l'algorithme $\mathcal{K}$ pour générer une paire de clés (p_k, s_k) en prenant pour entrée 1^k : $(p_k, s_k) \longleftarrow \mathcal{K}(1^k)$, une copie de la clé publique est envoyée à l'attaquant.

Etape 2 :

$\mathcal{A}_1$ prend en entrée la clé p_k, utilise éventuellement un oracle de déchiffrement $\mathcal{O}_1$ et sort la description d'un ensemble $\mathcal{M}$ de textes clairs (muni de la probabilité uniforme) ainsi que la description de l'état s du système (s peut représenter toutes les informations et outils dont l'attaquant dispose et souhaite utiliser dans $\mathcal{A}_2$).

Etape 3 :

Un oracle de chiffrement aléatoire $\mathcal{O}_c^*$ choisit un texte clair $m \in \mathcal{M}$ et le chiffre.

$\mathcal{O}_c^*$:

Entrée : $\mathcal{M}, p_k, R$

$m \longleftarrow \mathcal{M}$

$r \longleftarrow R$

$c \longleftarrow \mathcal{E}(p_k, m, r)$

sortie : c^* (=le challenge).

Etape 4 :

$\mathcal{A}_2$ prend en entrée $(\mathcal{M}, s, c^*)$ et sort une valeur z et la description d'une fonction $g : \mathcal{M} \longrightarrow \mathcal{M}$ qui évalue la relation entre m et z.

NB : l'attaquant souhaite que $g(m) = z$ avec une grande probabilité.

Modélisation des 4 étapes :

$Esp_{\Pi}^{SS}(\mathcal{A}, k)$, SS= Sécurité Sémantique.

$\widetilde{Esp}_{\Pi}^{SS}(\mathcal{A}, k)$, version aléatoire.

Version pratique		Version aléatoire
$(p_k, s_k) \longleftarrow \mathcal{K}(1^k)$	challenger	$(p_k, s_k) \longleftarrow \mathcal{K}(1^k)$
$(\mathcal{M}, s) \longleftarrow \mathcal{A}^{\mathcal{O}_1}(p_k)$	attaquant	$(\mathcal{M}, s) \longleftarrow \mathcal{A}^{\mathcal{O}_2}(p_k)$
$m \overset{rand}{\longleftarrow} \mathcal{M}$	challenger	$m, \widetilde{m} \overset{rand}{\longleftarrow} \mathcal{M}$
$r \overset{rand}{\longleftarrow} R$	challenger	$r \overset{rand}{\longleftarrow} R$
$c^* \longleftarrow \mathcal{E}(p_k, m, r)$	challenger	$c \longleftarrow \mathcal{E}(p_k, m, r)$
$(g, z) \longleftarrow \mathcal{A}_2(\mathcal{M}, s, c^*)$	attaquant	$(g, z) \overset{rand}{\longleftarrow} \mathcal{A}_2(\mathcal{M}, s, c)$
$\widetilde{z} \longleftarrow g(m)$ $si\ z = \widetilde{z}\ retourner\ 1$ $sinon\ retourner\ 0$	$\longrightarrow$ Résultats $\longleftarrow$	$\widetilde{z} \longleftarrow g(\widetilde{m})$ $si\ z = \widetilde{z}\ retourner 1$ $sinon\ retourner\ 0$

On compare les probabilités de succès dans ces deux scénarios qui doivent être indistinguables.

NB : Il existe une autre approche consistant à demander à l'attaquant de choisir entre deux scénarios dont un seul des deux aboutit à un succès.

b) **Avantage :**

L'avantage de l'attaquant est

$$Adv_{\Pi}^{SS}(A,k) = \left| Pr\left[Exp_{\Pi}^{SS}(A,k)=1\right] - Pr\left[\widetilde{Exp}_{\Pi}^{SS}(A,k)=1\right]\right|$$

c) **Définition de la sécurité sémantique :**

Le système de cryptographie à clé publique $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ est sémantiquement sûr contre une attaque du type CPA (respectivement CCA1, CCA2) si et seulement si pour tout polynôme $P(k)$ (où k est le paramètre de sécurité) et pour tout attaquant $\mathcal{A}$ de classe CPA (respectivement : CCA1, CCA2) dont les entrées et sorties se comportent comme dans $Exp_{\Pi}^{SS}(A,k)$ et $\widetilde{Exp}_{\Pi}^{SS}(A,k)$ et tel que :

$c(1)$ $\mathcal{A}$ s'exécute en temps $\leq P(k)$;

$c(2)$ l'échantillonnage de $\mathcal{M}$ s'exécute en temps $\leq P(k)$;

$c(3)$ la fonction g est calculable en temps $P(k)$;

$c(4)$ pour tout $m > 0$, $\lim k^m Adv_{\Pi}^{SS}(A,k) = 0$;

Autre approche équivalente :

"Pour extraire une information $g(m)$ d'un message m connaissant une autre information $h(m)$, on fait pas mieux si on connaît le chiffré c de m". Ce qu'on formalise par :

$$Adv_{\Pi}(\mathcal{A}) = \left| Pr\left[\mathcal{A}(h(x_k), \mathcal{E}(\mathcal{K}(1^k), x_k)) = g(x_k)\right] - Pr\left[\mathcal{A}(h(x_k)) = g(x_k)\right]\right|$$

est négligeable en k (=paramètre de sécurité) où x_k est une variable aléatoire sur l'ensemble des messages et g et h sont des fonctions d'extraction d'informations.

Indistinguabilité

a) **Modélisation :**

Etant donnés

- deux messages m_0 et m_1 ;
- c^* le chiffré de l'un des m_i (au hasard) ;

l'attaquant ne doit pas pouvoir distinguer lequel des m_i a été chiffré en c^*.

Ainsi la probabilité de réussite de l'attaquant doit être proche de celui qui aurait répondu au hasard.

Comme la réponse au hasard pour un message sur deux a pour probabilité 1/2 alors l'avantage de l'attaquant est $Adv = P(\mathcal{A} = \text{bonne réponse }) - 1/2$ ou $Adv = 2P(\mathcal{A} = \text{bonne réponse }) - 1$ car on s'intéresse à l'écart et non à la valeur de $P(\mathcal{A} = \text{bonne réponse})$.

Expérience :

$(p_k, s_k) \longleftarrow \mathcal{K}(1^k) :$ *challenger*
$(m_0, m_1, s) \longleftarrow \mathcal{A}^{\mathcal{O}_1}(p_k) :$ *attaquant*
$b \overset{rand}{\longleftarrow} \{0,1\} :$ *challenger*

$r \xleftarrow{rand} R :$ *challenger*

$c^* \longleftarrow \mathcal{E}(p_k, m_b, r) :$ *challenger*

$\widetilde{b} \longleftarrow \mathcal{A}_2(m_0, m_1, c^*, s) :$ *attaquant*

$$\left.\begin{array}{c} \widetilde{b} \in \{0,1\} \\ si\ b = \widetilde{b}\ retourner 1 \\ sinon\ retourner\ 0 \end{array}\right\} \text{ Résultats}$$

b) **Avantage :**

$$Adv_{\Pi}^{IND}(\mathcal{A}, k) = 2Pr\left[Exp_{\Pi}^{IND}(\mathcal{A}, k) = 1\right] - 1$$

On note souvent pour résumer

$$Adv_{\Pi}^{IND}(\mathcal{A}, k) = 2Pr\left[b = \widetilde{b}\right] - 1$$

c) **Définition :**

On dit que le système cryptographique Π est sûr au sens de l'indistinguabilité contre une attaque de type CPA (respectivement : CCA1, CCA2) si et seulement si pour tout attaquant $\mathcal{A}$ de la classe CPA (respectivement : CCA1, CCA2) dont les entrées et sorties se comportent comme il a été décrit dans $Exp_{\Pi}^{IND}(\mathcal{A}, k)$ et pour tout entier $m > 0$, nous avons $\lim\limits_{k \longrightarrow +\infty} Adv_{\Pi}^{IND}(\mathcal{A}, k) = 0$.

d) **Autre approche équivalente :**

On considère trois messages (m_0, m_1, c)

L'attaquant

- connaissant $\mathcal{E}(p_k, m_0, r_0) = c_0$, détermine la probabilité que c soit le chiffré de m_0 c'est à dire

$$Pr\left[c = \mathcal{E}(p_k, m_0, r_0')/\mathcal{E}(p_k, m_0, r_0)\right]$$

- connaisssant $\mathcal{E}(p_k, m_1, r_1) = c_1$, détermine

$$Pr\left[c = \mathcal{E}(p_k, m_1, r_1')/\mathcal{E}(p_k, m_1, r_1)\right]$$

Si ces deux probabilités sont suffisamment distinctes alors l'attaquant peut réussir à distinguer lequel des m_i $(i = 0\ ou\ 1)$ est chiffré en c.

On écrit alors les deux expériences suivantes.

Etant donné un triplet (m_0, m_1, c), l'attaquant $\mathcal{A}$ déroule les deux expériences :

$Exp_{\Pi}^{IND}(\mathcal{A}, k, (m_0, c))$	$Exp_{\Pi}^{IND}(\mathcal{A}, t, (m_1, c))$
$r_0 \longleftarrow R$	$r_1 \longleftarrow R$
$c_0 \longleftarrow \mathcal{E}(p_k, m_0, r_0)$	$c_1 \longleftarrow \mathcal{E}(p_k, m_1, r_1)$
$b \longleftarrow \mathcal{A}(c, c_0)$	$\widetilde{b} \longleftarrow \mathcal{A}(c, c_1)$
$b \in \{0,1\}$	$\widetilde{b} \in \{0,1\}$

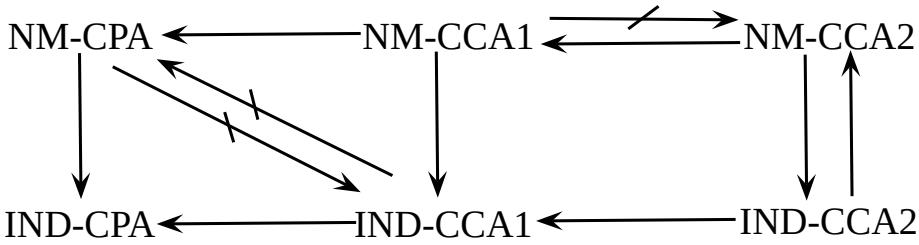
Donc pour qu'il y ait indistinguabilité il faut que $\forall m > 0$,

$$\lim k^m Pr\left[(m_0, m_1, c) / \left(Pr(\mathcal{A}(c, \mathcal{E}(p_k, m_0, r_0))) = 1\right) - \left(Pr(\mathcal{A}(c, \mathcal{E}(p_k, m_1, r_1))) = 1\right)\right] = 0$$

ou en terme de variables aléatoires

$$\lim k^m Pr\left[(X_n, Y_n, Z_n) / \left(Pr(\mathcal{A}(Z_n, \mathcal{E}(\mathcal{K}_{1^k}, X_n))) = 1\right) - \left(Pr(\mathcal{A}(Z_n, \mathcal{E}(\mathcal{K}_{1^k}, Y_n))) = 1\right)\right] = 0$$

2.3.5 Diagramme des relations entre les différentes notions de sécurité



2.3.6 Formalisme des preuves de sécurité des signatures numériques

Pour de plus amples informations voir l'article de J. Stern et D. Pointcheval [37] et l'article fondateur de Goldwasser, Micali et Rivest [18].

Signature

Une signature aléatoire est un triplet $(Genkey,\ Sig,\ Ver)$

• **Genkey :** prend en entrée un paramètre de sécurité k et sort un couple de clés publique/privée (p_k, s_k).

• **Sig :** algorithme de signature :

- prend en entrée un paramètre de sécurité k et fabrique une valeur aléatoire ρ.

- prend en entrée (s_k, m, ρ) où m est un texte à signer et produit une signature σ. [NB : il y a des variantes pour lesquelles ρ n'est pas donnée en sortie].

• **Vér :** algorithme de vérification :

prend en entrée (m, ρ, σ, p_k) ; l'algorithme de vérification sort 1 si la signature est valide et 0 sinon.

Modélisation

(1) **Unbreakability (UB) :** l'attaquant retrouve la clé privée (ou clé équivalente) à partir de la clé publique (cas de RSA).

(2) **Existential Unforgeability (EUF) :** l'attaquant crée un message et une signature valide.

(3) **Unforgeability against Adaptative Chosen Message Attack (EUF-CMA) :**

l'attaquant est autorisé à avoir accès à un oracle de signature (sans limite) et peut solliciter la signature de n'importe quel message (plusieurs sollicitations du même message sont possibles grâce à la valeur aléatoire) sans le challenge évidemment.

Chosen Message Security :

EUF-CMA est le niveau supérieur de sécurité pour les signatures (l'équivalent de CCA2 pour le chiffrement).

Une signature est dite (q, ε, τ)-sûre si pour tout attaquant $\mathcal{A}$ dont le temps de travail est majoré par τ, on ait :

$$Succ^{EUF-CMA}(\mathcal{A}, \mathcal{P}) = Pr\left(\left\{\begin{array}{l}(p_k, s_k) \longleftarrow Gen(1^k)\\(\mathcal{M}^*, \rho^*, \sigma^*) \longleftarrow \mathcal{A}^{S(S_k,.)}(p_k)\\Ver(p_k, \mathcal{M}^*, \rho^*, \sigma^*) = 1\\(\mathcal{M}^*, \rho^*) \notin Hist(S)\end{array}\right.\right) \leq \varepsilon$$

où :

- **Gen** est l'algorithme de génération de clés ;

- $\mathcal{A}^{S(S_k,.)}(p_k)$ signifie que l'attaquant $\mathcal{A}$ peut prendre en entrée la clé publique p_k et solliciter q appels à l'oracle de signature $S(S_k, .)$.

- **Hist(S)** est l'ensemble des couples $(\mathcal{M}, \rho)$ pour lesquels l'algorithme de signature $S(S_k, .)$ a produit une signature.

2.3.7 Exemple de preuve de sécurité dans le modèle standard

Signature Jetable de Groth (One Time Signature), (voir [19]) :

Soit g un générateur de $(\mathbb{Z}/p\mathbb{Z})^*$

- **GenKey :** $p_k = (X = g^x, Y = g^y, Z = g^z)$ clé publique.

- **Sig :** $m \in (\mathbb{Z}/p\mathbb{Z})^*,\ r \longleftarrow (\mathbb{Z}/p\mathbb{Z})^*$

Calculer $s = (1 - mx - yr)/z\ \in (\mathbb{Z}/p\mathbb{Z})^*$, retourner $\sigma = (r, s)$.

- **Ver :** $\sigma = (r, s)$ sur m

Calculer $X^m Y^r Z^s \stackrel{?}{=} g$

Théorème 2.3.1. *Si le logarithme discret est difficile dans G alors la signature jetable de Groth est sûre EUF-CMA dans le modèle standard.*

Preuve :

- On note $(g, h = g^\alpha)$ l'instance difficile à casser.

- On le cache dans la clé publique en prenant $X = g^{a_1}h^{b_1}$, $Y = g^{a_2}h^{b_2}$ *et* $Z = g^{a_3}$ où $a_1, b_1, a_2, b_2, a_3 \stackrel{Rand}{\longleftarrow} (\mathbb{Z}/p\mathbb{Z})^*$

- le propriétaire de la clé publique gère l'oracle de signature donc doit être capable de signer sans sa clé privée : si on donne m, il calcule $r = -mb_1/b_2 \mod p$ et $s = (1 - ma_1 - ra_2)/a_3 \mod p$ et on vérifie que $\sigma = (r, s)$ est une signature valide.

- Si l'attaquant fabrique une signature $\sigma_0 = (r_0, s_0)$ pour un texte m_0, alors on sait que

$r_0 = 1 - m_0 b_1/b_2$ et $s_0 = (1 - m_0 a_1 - r_0 a_2)/a_3$;

$X^{m_0} Y^{r_0} Z^{s_0} = g \Longleftrightarrow g^{m_0 a_1 + \alpha b_1 + a_2 + \alpha b_2 + s_0 a_3} = g \Longleftrightarrow m_0 a_1 + \alpha b_1 + a_2 + \alpha b_2 + s_0 a_3 = 1$

$\Longleftrightarrow \alpha(b_1 + b_2) = 1 - m_0 a_1 - s_0 a_3 \Longrightarrow \alpha = (1 - m_0 a_1 - s_0 a_3)/(b_1 + b_2)$ donc le logarithme discret de $h = g^\alpha$ est connu.

2.3.8 Exemple de preuve de sécurité sur le chiffrement

Problèmes DLP, CDH, DDH

Soit G un groupe admettant un sous groupe cyclique H d'ordre premier q assez grand et g un générateur de $H = \langle g \rangle$.

- **Le Problème du Logarithme Discret (DLP)**

Etant donné $y \in H$, calculer x tel que $y = g^x$. On pose alors $x = \log_g y$. Le succès d'un attaquant $\mathcal{A}$ est donné par $Succ^{DLP}(\mathcal{A}) = Pr[\mathcal{A}(g^x) = x]$.

- **Le Problème Calculatoire de Diffie-Hellman (CDH)**

Etant donné deux éléments $\alpha,\ \beta \in H$, $\alpha = g^a,\ \beta = g^b,\ calculer\ \gamma = g^{ab}$.

On définit $\gamma = CDH(\alpha, \beta)$. $Succ^{CDH}(\mathcal{A}) = Pr_{a,b \in \mathbb{Z}/q\mathbb{Z}}[\mathcal{A}(g^a, g^b) = g^{ab}]$

- **Le Problème Décisionnel de Diffie-Hellman (DDH)**

Etant donnés trois éléments $\alpha,\ \beta,\ \gamma \in H$. Décider si $\gamma = CDH(\alpha, \beta)$ i.e.

aussi si $\alpha = g^a,\ \beta = g^b,\ \gamma = g^c$, décider si $c = ab \mod q$.

L'algorithme qui résout DDH est appelé **distingueur** $\mathcal{D}$.

L'avantage est donné par :

$Adv^{DDH}(\mathcal{D}) = \left| Pr_{a,b,c \in \mathbb{Z}/q\mathbb{Z}} \left[1 \longleftarrow \mathcal{D}(g^a, g^b, g^c)\right] - Pr_{a,b \in \mathbb{Z}/q\mathbb{Z}} \left[1 \longleftarrow \mathcal{D}(g^a, g^b, g^{ab})\right] \right|$

Sécurité du chiffrement El Gamal en modèle standard

- **Système El Gamal :**

Soient p et q deux nombres premiers tel que q divise $p-1$. Soit h un générateur de $(\mathbb{Z}/q\mathbb{Z})^*$ et $g = h^{\frac{p-1}{q}}$ alors g est d'ordre q. On pose $H = \langle g \rangle$.

Clé privée de Bob : (x, p, q, g), $x \overset{Rand}{\longleftarrow} \mathbb{Z}/q\mathbb{Z}$.

Clé publique de Bob : (y, p, q, g), $y = g^x \mod p$.

Chiffrement : Alice choisit $m \in H$ puis $k \overset{Rand}{\longleftarrow} H$ et calcule $m_1 = g^k \mod p$ et $m_2 = y^k m \mod p$.

Le chiffré $c = (m_1, m_2)$.

Déchiffrement : $m = m_2 m_1^{-x} \mod p$

Théorème 2.3.2. *CDH $\Longleftrightarrow$ OW-CPA.*

L'inversion (OW-CPA) du chiffrement El Gamal est équivalente au Problème Diffie-Hellman Calculatoire c'est-à-dire pour un paramètre de sécurité t, s'il existe un attaquant $\mathcal{A}$ qui inverse El Gamal alors on peut construire un algorithme $\mathcal{B}$ qui résout CDH, $\left(Succ^{OW-CPA}(\mathcal{A}, t) \leq Succ^{CDH}(\mathcal{B}, t)\right)$:

Preuve :

Soit une instance aléatoire $(\alpha = g^a, \beta = g^b)$ du problème CDH que l'on veut résoudre, considérant l'attaquant $\mathcal{A}$ contre OW-CPA. On note Exp_0 le scénarios de l'attaquant, on le modifie progressivement pour résoudre CDH.

$Exp_0(\mathcal{A}, k)$

$x \overset{Rand}{\longleftarrow} \mathbb{Z}/q\mathbb{Z}$

$y = g^x \in H$

$(p_k, s_k) = (y, x)$

$m \overset{Rand}{\longleftarrow} H$

$k \longleftarrow \mathbb{Z}/q\mathbb{Z}$

$m_1 = g^k \mod p,\ m_2 = y^k m_2 \mod p$

$\widetilde{m} \longleftarrow \mathcal{A}(y, m_1, m_2)$

On note S_0 l'évènement "$m = \widetilde{m}$" et $Pr(S_0) = \varepsilon$

On modélise le jeu réel de façon successive :

$\underline{Exp_1(\mathcal{A}, k)}$

Dans Exp_0, on remplace $(x, y = g^x)$ par $(a, \alpha = g^a)$, cela est possible car $\alpha = g^a, \beta = g^b$ sont aléatoires.

On note $S_1 =$ "$m = \widetilde{m}$" et on a $Pr(S_1) = Pr(S_0)$ car a et g^a sont aléatoires.

$\underline{Exp_2(\mathcal{A}, k)}$

Dans Exp_1 , on remplace $m_1 = g^k$ par $m_1 = g^b = \beta$ et on note $S_2 =$ "$m = \widetilde{m}$" alors $Pr(S_2) = Pr(S_1)$ car β est aléatoire.

$\underline{Exp_3(\mathcal{A}, k)}$

Dans Exp_2, on remplace $m_2 = my^k$ où $m \overset{Rand}{\longleftarrow} H$ par $m_2 \overset{Rand}{\longleftarrow} H$ (aléatoire, créé directement)

La structure de groupe fait que la distribution est uniforme pour les 2 facteurs, on note $S_3 =$ "$Sm_1^{-x} = m$"

$Pr(S_3) = Pr(S_1)$

Récapitulation :

On a maintenant

$\varepsilon = Pr(S_0) = Pr(S_3)$

$$= Pr\left[\mathcal{A}(y, m_1, m_2) = m = S/m_1^x : \left\{ \begin{array}{l} S \overset{Rand}{\longleftarrow} H,\ y = g^a \\ a, b \longleftarrow \mathbb{Z}/q\mathbb{Z},\ m_1 = g^b \end{array} \right.\right]$$

$$= Pr\left[\mathcal{A}(y, m_1, m_2) = S/CHD(\alpha, \beta) : \left\{ \begin{array}{l} \alpha, \beta, S \longleftarrow H \\ y = \alpha,\ m_1 = \beta \end{array} \right.\right]$$

car si $m = S/m_1^x$ alors $g^{ab} = S/m$.

NB1 : (très important)

Il n'y a pas de non malléabilité pour le chiffrement El Gamal à cause de la propriété d'homomorphisme. Mais seulement la sécurité sémantique, c'est l'une des raisons qui fait qu'on utilise pas le chiffrement de El Gamal.

NB2 :

Dans le chiffrement El Gamal, on peut prendre $m \in G = (\mathbb{Z}/p\mathbb{Z})^*$ et $m \notin H = \langle g \rangle$ cela n'empêche pas le déchiffrement. Mais l'Exp_3 de la preuve de sécurité, montre que dans ce cas la distribution n'est pas uniforme car $m \notin H$ et $y^k \in H$ et donc $Pr(S_3) \neq Pr(S_1)$ ce qui fausse la preuve de sécurité.

Théorème 2.3.3. *La sécurité sémantique IND-CPA du chiffrement El Gamal est équivalente au problème Diffie-Hellman Décisionnel.* $Adv^{IND-CPA}(\mathcal{A}, t) \leq 2 \times Adv^{DDH}(\mathcal{D}, t)$ *où* $\mathcal{A}$ *est un attaquant contre El Gamal et* $\mathcal{D}$ *un distingueur Diffie-Hellman.*

Preuve :

Soit $(\alpha = g^a, \beta = g^b)$ une instance aléatoire. Notons $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ un attaquant contre le système El Gamal en temps t.

$\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

Exp_0 :

$(y = g^x, x \overset{Rand}{\longleftarrow} \mathbb{Z}/q\mathbb{Z}) \longleftarrow \mathcal{K}(1^k)$

$(m_0, m_1) \longleftarrow \mathcal{A}_1^{\mathcal{D}}(y)$

$\delta \overset{Rand}{\longleftarrow} \{0, 1\}$

$(\widetilde{m}_1, \widetilde{m}_2) \longleftarrow \mathcal{E}(y, m_\delta)$

$\{0, 1\} \ni \delta' \longleftarrow \mathcal{A}_2^{\mathcal{D}}(\widetilde{m}_1, \widetilde{m}_2)$

On note $\varepsilon' = P(\delta = \delta') = P(S_0)$.

Exp_1 :

On modifie $(y, \widetilde{m}_1)$ en (α, β), $\widetilde{m}_2$ reste inchangé (donc $x = a,\ k = b$).

Les distributions de $x,\ y$ sont identiques car (α, β, a) est aléatoire.

On note S_1 l'évènement associé. On a $P(S_1) = P(S_0)$ car (α, β) est aléatoire.

Exp_2 :

On modifie $\widetilde{m}_2$ dans le cas précédent au lieu de $\widetilde{m}_2 = m_\delta y^k$ on pose $\widetilde{m}_2 = m_\delta C$ où $C = DH(A, B)$. Si S_2 est l'évènement aléatoire associé alors $P(S_2) = P(S_1)$.

Exp_3 :

Ici on remplace $C = DH(A, B)$ par $C = g^c$, c aléatoire.

On note S_3 l'évènement associé.

Maintenant, étant donné que $\delta' = \delta$ est un évènement détectable, on peut définir un algorithme distingueur qui :

- exécute le jeu Exp_2 si $C = DH(A, B)$
- exécute le jeu Exp_2 si $C \neq DH(A, B)$ et qui
  - retourne 1 si $\delta' = \delta$
  - retourne 0 si $\delta' \neq \delta$

Mais, on a

$Pr[1 \longleftarrow \mathcal{D}/C \overset{Rand}{\longleftarrow} \langle g \rangle] = Pr[S_2]$

$Pr[1 \longleftarrow \mathcal{D}/C = DH(A, B)] = Pr[S_3]$

En considérant que DH(A, B) et C pour Exp_3 sont tous les deux parfaitement aléatoires, on voit que $Pr[S_3] = 1/2$

en posant $\varepsilon' = \frac{1+\varepsilon}{2}$, on a $Adv^{DDH}(t) \geq Pr[S_2] - Pr[S_3] = \frac{1+\varepsilon}{2} - \frac{1}{2} = \varepsilon$

par suite $\varepsilon \leq 2Adv^{DDH}(t)$ d'où l'avantage recherché.

Chapitre 3

Protocoles d'échange de clés

3.1 Généralités sur les protocoles

Les modes communication ont connu de grands changements (téléphone, courrier électronique, etc...) qui sont difficiles à maîtriser du point de vu de la sécurité.

Les nouveaux systèmes de communication sont beaucoup plus rapides que l'intervention humaine et moins contraignants mais n'offrent pas les propriétés de sécurité nécessaires à toute communication importante.

C'est ainsi que les protocoles de communication ont été développés pour combler ces déficits de sécurité, donc d'empêcher la possibilité pour un participant de la communication ou une personne extérieure d'obtenir une information qu'il ne devrait pas connaître, ou de se faire passer pour quelqu'un d'autre.

3.2 Protocoles cryptographiques

3.2.1 Protocoles de communication

a) Échange de messages

La communication est l'échange d'information entre deux ou plusieurs personnes via un canal qui peut être public (les médias, tels la presse ou la radio) ou privé ; chaque personne étant appelée participant ou agent. Ces participants communiquent grâce à des envois de messages (concrètement, chaque message envoyé n'est qu'une suite de bits).

Un protocole de communication permet l'échange de messages entre différents participants via un canal de communication. La forme, le contenu des messages, les différents participants à la communication et l'ordre dans lequel les messages sont échangés via un canal de communication spécifient un protocole de communication.

b) Session

L'étude des instances d'un protocole cryptographique introduit la notion de session de protocole qui est un ensemble d'échanges de messages entre plusieurs participants formant un ensemble cohérent et pouvant être répété. Pour différencier deux sessions différentes, on introduit généralement dans le protocole des nombres aléatoires générés par les participants de la communication, appelés nonces. Une nonce est une donnée de grande taille, choisie au hasard par un participant et utilisée une seule fois.

c) Les participants

Chaque participant peut établir, simultanément, avec différents participants, plusieurs sessions du protocole. Les participants sont de deux types : honnêtes ou intrus.

Participants honnêtes

Dans la description d'un protocole, les participants sont supposés être honnêtes, c'est-à-dire des participants qui ne collaborent pas avec les attaquants et qui effectuent correctement les échanges dans l'ordre défini par le protocole.

Pour réaliser correctement un protocole, il faut des participants honnêtes pour respecter les spécifications du protocole.

Intrus

Un intrus ou attaquant est un participant qui ne suit pas exactement le déroulement du protocole. Il espionne les communications qui circulent sur les canaux publics, joue plusieurs sessions de protocoles avec des participants, en se faisant passer pour un agent honnête et ainsi effectue des actions non prévues par la spécification du protocole, afin de découvrir des informations supposées rester secrètes.

3.2.2 Vérification de protocoles cryptographiques

Actuellement, de plus en plus de protocoles cryptographiques sont vérifiés avant leurs commercialisations. L'analyse de ces protocoles est un problème de vérification en présence de canaux de communications non sécurisés.

Pour attaquer un protocole cryptographique, il existe deux approches possibles : La première consiste à essayer de déchiffrer les messages chiffrés échangés. Ce type d'attaques développées par les cryptographes porte sur l'algorithme cryptographique employé. La seconde approche suppose que la méthode de chiffrement est inviolable, grâce à un raisonnement logique, cherche à obtenir de l'information.

Nous énonçons ici les principales propriétés concernant les protocoles cryptographiques.

Propriétés à vérifier

Il existe de nombreuses propriétés de sécurité, nous présentons en quelques mots les plus usuelles.

1. **Secret :** La notion de secret s est utilisée par la plupart des auteurs. Un secret s est une donnée confidentielle ne devant pas être découverte par une tierce personne qui n'est pas censée la connaître. Un protocole vérifie la propriété de secret pour un secret s, si une personne malhonnête (l'intrus) en fonction de ses capacités ne peut jamais obtenir une partie de s échangé entre plusieurs participants honnêtes.
2. **Accord non répudiable :** Un protocole doit établir un accord non répudiable entre deux agents c'est à dire chaque agent peut fournir la preuve que l'autre a accepté les termes de l'accord.
3. **Authentification :** La propriété d'authentification garantit à un interlocuteur qu'il communique avec le "vrai" participant. Ceci est fort utile pour sécuriser les transactions bancaires sur internet par exemple.
4. **Équité :** Un protocole d'accord non répudiable entre deux agents A et B doit être équitable c'est à dire aucun agent ne peut obtenir d'avantage sur l'autre : A n'obtient pas la preuve de l'accord de B avant que B n'ait une preuve de l'accord de A (et vice-versa).

3.3 Présentation de quelques protocoles

3.3.1 Le protocole de Needham-Schroeder

Le protocole (1978, R. Needham et M. Schroeder [33]) décrit l'échange de messages entre deux participants, par exemple Alice et Bob, que nous dénoterons respectivement par A et B. Le protocole utilise un algorithme de chiffrement asymétrique. Nous dénotons par $\{m\}_{pub(A)}$ un message m chiffré par la clé publique d'Alice $pub(A)$ et considérons que tous les participants connaissent toutes les clés publiques. Le protocole est décrit comme suit :

$$\begin{aligned} &1.\ A \longrightarrow B : && \{A, N_a\}_{pub(B)} \\ &2.\ B \longrightarrow A : && \{N_a, N_b\}_{pub(A)} \\ &3.\ A \longrightarrow B : && \{N_b\}_{pub(B)} \end{aligned}$$

Une fois le protocole terminé, Alice est convaincue d'avoir effectué une session du protocole avec Bob, car il lui a renvoyé sa nonce. De même, Bob pense avoir communiqué avec Alice. Ce protocole peut être cryptanalysé.

3.3.2 Le protocole HMQV

Le protocole MQV [29] de Law, Menezes, Qu, Solinas et Vanstone est connu comme étant l'un des protocoles d'échange de clé (basés sur celui de Diffie-Hellman) le plus efficient qui utilise une

authentification basée sur une clé publique. En plus d'une grande performance, le protocole a été fait pour vérifier plusieurs propriétés de sécurité. MQV présente néanmoins des vulnérabilités.

Le protocole HMQV est présenté dans [28] comme une variante minutieusement modifiée de MQV, qui produit les mêmes performance et fonctionnalités du protocole original mais la plupart des buts de sécurité de MQV peuvent être formellement prouvés dans le modèle de l'oracle aléatoire sous l'hypothèse de Diffie-Hellman calculatoire. HMQV est une nouvelle forme de "challenge-response signatures", dérivée du schéma d'identification de Schnorr, qui a la propriété que le challenger et le signataire peuvent calculer la même signature ; le premier en ayant choisi le challenge et le deuxième en connaissant la clé privée de la signature.

Protocole

On se donne une fonction de hachage H et un groupe cyclique G de générateur g ($q = \#G$).

On suppose qu'Alice (respectivement : Bob) a une paire de clés privée/publique $(\alpha_A, \gamma_A = g^{\alpha_A})$ $\big($respectivement : $(\alpha_B, \gamma_B = g^{\alpha_B})\big)$ où α_A, α_B sont des entiers aléatoires avec $\alpha_A, \alpha_B < q$.

1. Alice choisit un aléa de session secret $x_{As} < q$, calcule $X_A = g^{x_A}$ et envoie X_A à Bob ;
2. Bob choisit un aléa de session secret $x_{Bs} < q$, calcule $X_B = g^{x_B}$ et envoie X_B à Alice ;
3. Alice calcule le paramètre $e_s = H\big(X_B, id_A\big)$ et la clé $g_{Ks} = \big(X_B \gamma_B^{e_s}\big)^{x_A + d_s \alpha_A}$;
4. Bob calcule le paramètre $d_s = H\big(X_A, id_B\big)$ et la clé $g_{Ks} = \big(X_A \gamma_A^{d_s}\big)^{x_B + e_s \alpha_B}$;
5. Alice et Bob calculent ensemble la clé secrète commune $K = \overline{H}(g_{Ks})$.

Sécurité et performance : HMQV est une amélioration du protocole MQV. C'est un protocole sûr avec une grande performance, mais il est vulnérable à l'attaque PFS et ne produit pas l'authentification mutuelle avec les clés privée et publique entre les parties qui communiquent d'où l'établissement de la session n'échoue jamais. Néanmoins, il reste l'un des meilleurs protocoles d'échange de clés à ce jour.

3.3.3 Identification de Lamport

Bob dispose d'une paire de clés publique/privée $(pub(B), priv(B))$. Si T est donné, alors $\{T\}_{pub(B)}$ signifie que la donnée a été chiffrée via $pub(B)$ et l'algorithme asymétrique associé.

Alice souhaite s'identifier $t-1$ fois successivement auprès de Bob.

Protocole :

1. **Etape 1** : Alice choisit un bon mot de passe W (secret !) et une fonction de hachage publique h et elle calcule $W_0 = h^t(W) = h \circ h \circ ....h(W)$ puis $A \to B : \{W_0 = h^t(W)\}_{pub(B)}$
2. **Etape 2** : Si Alice veut s'authentifier une $i$ème fois, elle envoie à Bob $A \to B$: $\{W_i = h^{t-i}(W)\}_{pub(B)}$
3. **Etape 3** Bob calcule $h(W_i) = h^{t-(i-1)}(W)$ et compare avec la dernière valeur qu'Alice lui a envoyé.

   Ensuite Bob détruit W_{i-1} puis garde W_i en prévision d'une future demande d'identification d'Alice

-

CONTRIBUTIONS

Résumé

Pour la cryptographie à clé secrète, l'échange de clé peut être assuré par des protocoles qui utilisent l'idée du protocole Diffie-Hellman et ce dernier repose sur le logarithme discret.

La cryptographie à clé publique joue un rôle fondamental pour la sécurité car elle offre des services que n'offre pas la cryptographie à clé secrète telle que la non répudiation entre autre. En cryptographie à clé publique, beaucoup de schémas utilisent le problème du logarithme discret dont le meilleur cadre d'utilisation actuel est le domaine des courbes elliptiques issues de la théorie des nombres.

Dans cette thèse nous avons contribué sur trois points essentiels :

- Dans le chapitre 4, nous avons proposé une modification du chiffrement et de la signature d' El Gamal, en permettant de réduire la taille des clés. Ce travail s'implémente facilement sur les courbes elliptiques ;

- Dans le chapitre 5, nous avons cryptanalisé le protocole d'échange de clé "Strong Diffie-Hellman DSA Key Exchange", puis nous l'avons amélioré même si notre proposition reste moins efficient que le protocole HMQV, (notre protocole peut être implémenté sur les courbes elliptiques).

- Dans le chapitre 6, nous avons proposé une nouvelle forme de courbe elliptique binaire, mais, nous n'avons pas pour le moment prouvé qu'elle soit adaptée complètement pour la cryptographie à clé publique.

abstract

In secret key cryptographic, key exchange can be insured by protocols which use Diffie-Hellman protocol idea and this one rest on discrete logarithm.

Public key cryptographic operate a fundamental function for security because it offers services that doesn't offer by secret key cryptography for example non-repudiation. In public key cryptographic, many of schemes use the discrete logarithm problem of which the best context actual of use is the elliptic curves domain born in number theory.

In this thesis we have contribute on three essential points :

- in chapter 4, in section 1, we describe the "Modified El Gamal Encryption Scheme" and after we study it's security and it's efficiency.

- in chapter 5, we do a cryptanalyse of the so called "Strong Diffie-Hellman-DSA Key Exchange (briefly : SDH-DSA-KE)" and after we propose "Strong Diffie-Hellman-Exponential-Schnnor Key Exchange (briefly : SDH-XS-KE)" which is an improvement for efficiency and security.

- in chapter 6, we have proposed a new binary elliptic curve. But, we have not proved for the present that is adapted in public key cryptographic.

Chapitre 4

Une nouvelle variante des schémas de chiffrement et de signatures El Gamal

Article publié :

" Demba SOW and Djiby SOW : *A new variant of El Gamal's encryption and signatures schemes*, (**JP Journal of Algebra, Number Theory and Applications Volume 20, Number 1, 2011, Pages 21-39**)"

Dans ce chapitre, les cryptosystèmes proposés sont une légère modification du DSA et des schémas d'El Gamal [17]. Toutefois, il n'est pas nécessaire de considérer le générateur et son ordre publiques. On peut utiliser une clé de déchiffrement plus petite que celle dans le schéma de El Gamal.

En général, si on travaille dans un sous-groupe cyclique de taille d (avec d un grand nombre premier), alors on peut garder d secret ; on peut aussi utiliser un exposant secret r pour le déchiffrement de taille $\frac{|d|}{n_0}$, (où n_0 est un entier qui divise $|d|$, la taille de d). Par exemple, il est possible, pour différents niveaux de sécurité, d'utiliser des clés de 160, 190 ou 256 bits pour le déchiffrement. Par conséquent, le nouveau schéma de chiffrement est plus rapide que celui classique d'El Gamal pour le processus de déchiffrement.

Nos variantes de schémas de signature sont plus sûres dans le sens que certaines vulnérabilités sur les schémas de signature Meta-El Gamal ne marchent pas avec les nouvelles modifications proposées. De plus, il existe beaucoup plus de variantes pour nos signatures que celles des "schémas de signature El Gamal".

Comme le schéma de chiffrement El Gamal, notre schéma de chiffrement est basé sur le problème DDH (Decisional Diffie-Hellman). De plus, le secret de l'ordre d et du générateur g (qui est optionnel) est basé sur le Problème de factorisation des entiers.

4.1 Présentation de l'algorithme

Il est bien connu qu'il existe très peu de cryptosystèmes à clé publique robustes et efficients.

Vu le rôle essentiel de la cryptographie à clé publique pour la sécurité de l'information, il est toujours intéressant de trouver un nouveau schéma asymétrique ou d'en améliorer un.

Fort de cette remarque, nous présentons un nouveau schéma à clé publique. Ce système peut être vu comme une modification du schéma de chiffrement El Gamal. Pour les applications, nous pouvons utiliser tout groupe qui a un sous-groupe cyclique avec un ordre suffisamment grand.

Par exemple, il est possible d'utiliser les groupes d'El Gamal avec un ordre connu ou inconnu tels que les groupes cycliques, les corps finis ou les courbes elliptiques sur les corps finis.

Notre algorithme a une version pour le chiffrement et une version pour la signature.

Ce nouveau cryptosystème est plus rapide que celui d'El Gamal et ainsi, semble être plus adapté pour la cryptographie embarquée telle que les cartes à puce.

En général, avec un sous-groupe cyclique d'ordre d (avec d un grand nombre premier), nous pouvons utiliser un exposant secret de taille arbitraire de la forme $\frac{|d|}{n_0}$ pour le déchiffrement et la signature (où n_0 est un entier et $|d|$ est la taille de d). Par exemple, nous pouvons utiliser une clé de 80, 128, 160, 190 ou 256 bits pour le déchiffrement et la signature pour toute taille de d pourvu que le logarithme discret reste difficile.

Maintenant nous donnons quelques notions et notations utilisées dans la suite.

Partout dans ce chapitre, pour un mécanisme de cryptosystème : Charlie est l'attaquant ; Bob est le récepteur et Alice est l'expéditrice dans le processus de chiffrement ; Alice est la vérificatrice et Bob effectue la signature pour le mécanisme de signature.

Les notations suivantes sont utilisées dans ce chapitre.

Si G est un groupe (noté multiplicativement), l'ordre de G est son cardinal et est noté $\#G$. l'ordre d'un élément $m \in G$ est noté par $o(m)$ est le plus petit entier u tel que $m^u = 1$. Si x est un nombre réel, $\lfloor x \rfloor$ représente la partie entière c'est-à-dire l'unique entier tel que $\lfloor x \rfloor \leq x < \lfloor x \rfloor + 1$. Si n est un entier, $|n|$ représente la taille de n en bits et $(\frac{\mathbb{Z}}{n\mathbb{Z}})^*$ est le groupe multiplicatif d'unités de l'anneau des entiers modulo n.

Ce chapitre est organisé comme suit :

- Dans la première section, nous décrivons le "schéma modifié du chiffrement El Gamal" et après nous étudions sa sécurité et son efficience.

- Dans la seconde section, quelques "schémas et variantes modifiés de la signature Meta-El Gamal" sont présentés avec une étude de leur sécurité et efficience.

- Dans la troisième section, une étude comparative par rapport à El Gamal est effectuée.

4.2 Chiffrement et déchiffrement

Dans cette section, on donne les algorithmes de génération de clé, de chiffrement et de déchiffrement qui peuvent être vus comme une légère modification des schémas d'El Gamal (voir [17] et [30]).

4.2.1 Algorithme de génération des clés

Pour créer une paire de clés, Bob devra faire ce qui suit :

- choisir un groupe G avec un ordre suffisamment grand : $n = \#G$; choisir un élément de groupe $g \in G$ avec un ordre premier suffisamment grand : $d = O(g)$ [d et g peuvent être optionnellement secrets, mais cela n'est nullement nécessaire pour la preuve de sécurité] ;
- choisir deux nombres premiers aléatoires $2 < k < r < d$ suffisamment grands, r et k premiers avec d et calculer kd ;
- calculer avec l'algorithme de division euclidienne le couple (s, t) tel que $kd = rs + t$ où $t = kd \mod r$ et $s = \lfloor \frac{kd}{r} \rfloor$; [Noter que $s = \lfloor \frac{kd}{r} \rfloor < d$, $\lfloor \frac{r}{t} \rfloor$, $d - s$ et t doivent être suffisamment grands, sinon retourner à l'étape 2] ;
- calculer $\gamma = g^s$ et $\delta = g^t$ dans G ; [Noter que $\gamma \neq 1$ et $\delta \neq 1$] ;
- la clé publique de Bob est $((\gamma, \delta), G, n = \#G)$ et sa clé privée est $((r, s, t), G, n = \#G)$.

Remarque 4.2.1. *Noter que seul r est utilisé dans le processus de déchiffrement. Mais (r, s, t) est utilisé dans le mécanisme de signature avec quelques conditions supplémentaires.*

4.2.2 Algorithme de chiffrement

Pour chiffrer un message pour Bob, Alice devra faire comme suit :

- prendre $((\gamma, \delta), G, \#G)$, la clé publique de Bob ;
- choisir un entier aléatoire $2 < \alpha < \#G$ tel que α et $\#G$ soient premiers entre eux ; calculer $m_1 = \gamma^\alpha$ et $\lambda = \delta^\alpha$ dans G, (d'où $m_1 \neq 1$ et $\lambda \neq 1$) ;
- transformer le message m comme un élément de G ;
- calculer $m_2 = \lambda m$ dans G ;
- le texte chiffré est (m_1, m_2).

4.2.3 Algorithme de déchiffrement

Pour déchiffrer un message chiffré avec sa clé publique, Bob devra faire comme suit :

- prendre (r, G), sa clé privée (première partie de la clé privée) ;
- prendre (m_1, m_2) le texte chiffré ;
- calculer m_1^r et $m_1^r m_2$ dans G ;
- le texte clair est $m_1^r m_2$.

Pourquoi le déchiffrement fonctionne ?

Proposition 4.2.1. *La fonction de déchiffrement :*

$$\begin{array}{rccc} D^{(r)} : & G \times G & \longrightarrow & G \\ & (m', m'') & \longmapsto & m'^r m'' \end{array}$$

est l'inverse à gauche de l'algorithme de chiffrement probabiliste qui est la fonction :

$$\begin{array}{rccc} E^{(g^s, g^t)}(\alpha, .) : & G & \longrightarrow & G \times G \\ & m & \longmapsto & (g^{\alpha s}, g^{\alpha t} m) = (g^s, g^t)^\alpha (1, m) \end{array}$$

Preuve : Nous avons : $D^r \circ E^{(g^s,g^t)}(\alpha, m) = D^r(g^{\alpha s}, g^{\alpha t}m) = g^{\alpha rs}g^{\alpha t}m = g^{\alpha(rs+t)}m = g^{\alpha(kd)}m = m$. D'où $D^{(r)}$ est l'inverse à gauche de $E^{(g^s,g^t)}(\alpha, .)$ qui signifie que le déchiffrement marche. □

Remarque 4.2.2. *Comme le schéma de chiffrement El Gamal, notre algorithme a la propriété d'homomorphie (voir [40] pour les définitions, les applications et les inconvénients pour les propriétés d'homomorphie des algorithmes de cryptographie).*

4.2.4 Implémentation dans $\mathbb{Z}/p\mathbb{Z}$, p premier

GROUPE, ORDRE ET GENERATEUR

$p = 709295319570854972016289429668809949256 9$

$d = 29553971648785623834012059569533747885 7$, d divise $p - 1$.

$g = 1689154080$, g est d'ordre d.

$k = 2996711289$

CLE PRIVEE

$s = 3754455090$

$r = 23589234216862565290211368881805498639 1$

$t = 36483$

CLE PUBLIQUE

$\gamma = g^s \mod p = 52063513865303382138579093468361028791 08$

$\delta = g^t \mod p = 49672803479916540323931028487107668096 04$

Texte à chiffrer : $m =$ "$salut$"

m en entier : $m = 495555736948$

CHIFFREMENT

Paramètre aléatoire $\alpha = 2718774999$

Le chiffré $m_1 = 59777023498094130380223296628197213772 60$

Le chiffré $m_2 = 29039854140336004179663172789278635898 68$

DECHIFFREMENT

Le déchiffré en entier : $m_1^r m_2 = 495555736948$

Le déchiffré en Chaine : $m =$ "$salut$"

Pour les codes en java, voir annexe.

4.2.5 Sécurité

(1) Preuve de sécurité. Posons $\gamma = g^s$, comme d est premier et s est primier avec d, alors γ génère le même sous groupe que g . En remarquant que $g^t = \gamma^\omega$ où $\omega = d - r$ est la clé secrète, on conclut que notre algorithme, transformé en un schéma El Gamal sous cette forme, a la même sécurité que ce dernier et donc est basé sur le problème de Décision de Diffie-Hellman (voir Boneh [7] et Joux [1]) .

(2) Sur la taille des paramètres. Avec les deux parties g^s et g^t de la clé publique, on peut essayer de chercher le premier entier w_0 tel que $g^t.(g^s)^{w_0} = 1$, dans G. Alors il existe w tel que $t + w_0 s = wd$. Mais $t + rs = kd$, ainsi $(k - w)d = (r - w_0)s$. Comme s et d sont copremiers et $w_0 \leq r \leq d$, alors $r = w_0$.

Ainsi, l'attaquant Charlie peut utiliser w_0 pour déchiffrer les messages chiffrés avec la clé publique d'Alice. D'où, dans le but de parer à cette vulnérabilité, il est nécessaire pour $r = \lfloor \frac{kd}{s} \rfloor$, d'être suffisamment large.

Remarque 4.2.3. *Noter que $g^t = (g^s)^{d-r}$ ainsi, il est important que $d - r$ soit suffisamment grand, c'est à dire $|d - r| \simeq |d|$.*

On peut essayer de chercher le premier entier w_0' tel que $(g^t)^{w_0'} = g^s$ dans G. Comme $t < s < d$, alors $tw_0' = s$. Ainsi, il est important que $\lfloor \frac{s}{t} \rfloor$ et t soient suffisamment grands.

4.2.6 Performance

Pour le déchiffrement, on peut utiliser le premier paramètre secret r et on calcule successivement m_1^r et $m_1^r m_2$. Le processus est rapide parce que, pour toute taille du sous groupe cyclique qui est utilisé, on peut choisir r petit (relativement au niveau de sécurité désiré) que celui utilisé dans le processus de déchiffrement El Gamal. Pour voir cela, on remarque que, si a est utilisé comme paramètre dans les schémas de chiffrement El Gamal avec un modulo premier d alors, dans le processus de déchiffrement, $d - a$ est utilisé comme exposant. Mais, l'exposant secret qui est utilisé dans le processus de déchiffrement $d - a$ reste très grand si d est grand. Maintenant, l'exposant secret s (utilisé dans l'algorithme de génération de clé) et r (utilisé comme exposant dans le processus de déchiffrement) vérifient la formule $kd = rs + t$, avec $t < s < d$ et $k < s < d$ alors $|k| + |d| \simeq |r| + |s|$, ainsi, on peut, en général, choisir $|s| \simeq \frac{(n_0-1)|d|}{n_0} + |k|$ et $|r| \simeq \frac{|d|}{n_0}$, où n_0 est un entier fixé. D'où, dans toute situation, on peut choisir s tel que r soit petit (mais suffisamment grand dans le but de parer aux attaques connues) et alors le processus de déchiffrement est toujours rapide. Mais le mécanisme de chiffrement a la même efficience que le mécanisme de chiffrement d'El Gamal.

4.2.7 Le cas où $d = o(g)$ est non secret

Dans la suite, nous donnons quelques suggestions pour la taille des paramètres quand notre cyptosystème est conçu dans $(\frac{\mathbb{Z}}{p\mathbb{Z}})^*$ où p est premier et dans le cas où d est non secret.

- p et d doivent être deux entiers premiers aléatoires très grands ("strong") avec $|p| \geq 1024$, $|d| \geq 512$ et d divise $p-1$;
- les entiers k, s, α et t doivent être choisis tels que : $|k| \simeq |\alpha| \geq 160$, $|s| \geq 160$ et $|r| \geq 160$ (avec $|d-r| \simeq |d|$ et $|s|-|t| \geq 160$) . Comme $|k|+|d| \simeq |r|+|s|$, pour **s** (utilisé comme exposant dans notre algorithme de génération de clés) et **r** (utilisé comme exposant dans notre processus de déchiffrement), nous pouvons en général choisir $|s| \simeq \frac{(n_0-1)|d|}{n_0} + |k|$ et $|r| \simeq \frac{|d|}{n_0}$, où n_0 est entier fixé.

Par exemples :

– si nous choisissons $|s| \simeq \frac{|d|}{2} + |k| \simeq 446$, avec $|k| \simeq 190$ et $|d| \simeq 512$, nous avons $|r| \simeq 256$;

– et si nous choisissons $|s| \simeq \frac{3|d|}{4} + |k| \simeq 384$, avec $|k| \simeq 128$ et $|d| \simeq 512$, nous avons $|r| \simeq 128$;

Remarque 4.2.4. ***Implémentation dans $\frac{Z}{nZ}$ où $d = o(g)$ est non secret***

Soit $n = p'q'$ un module RSA, ainsi nous pouvons implémenter notre schéma dans $\frac{Z}{nZ}$ comme précédent en utilisant $\varphi(n) = (p'-1)(q'-1)$ au lieu de $p-1$.

4.2.8 Le cas où $d = o(g)$ est secret

Dans cette section, nous voyons comment garder $d = o(g)$ secret (et d'où le générateur g directement utilisé, peut ne pas être publié même si $h = g^s$ est un autre générateur publique).

Si nous voulons garder d secret, il suffira de choisir un groupe G avec un ordre $n = \#G$ (non secret) tel que :

- il existe un entier premier aléatoire assez grand d qui divise $\#G = n$;
- et que le problème de factorisation est difficile pour $\#G = n$.

Noter que, si l'ordre d du générateur g est secret, il est difficile de décider ou de tester quand un élément g' sélectionné aléatoirement dans G, est un générateur du sous groupe cyclique $H =< g >$ de G (même si g^s est générateur publique de H). Dans [39], Poupard et Stern ont étudié des cas divers de probabilité pour trouver des générateurs de quelques sous groupes dans un groupe fini.

Implémentation dans $\frac{Z}{pZ}$ où $d = o(g)$ est secret

Par exemple, rendre secret d dans $\frac{Z}{pZ}$, on peut faire ce qui suit :

- sélectionner deux entiers aléatoires premiers très grands d et d' tels que $d-1$ et $d'-1$ soient des entiers non friables (c'est-à-dire sont divisibles par un grand nombre premier), et que le problème de factorisation est calculatoirement difficile pour dd' ;
- trouver un entier premier très grand p tel que dd' divise $p-1$.

Il est possible d'adapter légèrement l'algorithme de Gordon (voir Menezes et *al.* [30]) pour fabriquer des entiers premiers très grands tels que les entiers d et p que l'on veut ici.

NB : un nombre premier d est dit friable si $d-1$ n'a que de petits facteurs premiers.

Algorithme 4.2.1. *L'algorithme de Gordon modifié suivant [30] génère un nombre premier très grand p tel que $p-1$ est divisible par dd' où d et d' sont des entiers premiers aléatoires très grands et, $d-1$ et $d'-1$ sont entiers non friables.*

Algorithme :

(1) *générer trois nombres premiers aléatoires très grands* s'' , t' *et* $t"$ *avec* $|t'| \simeq |t"| \simeq 2|s"|$;

(2) (a) *sélectionner un entier* k'_0. *Trouver le premier entier premier dans la suite* $2k's" + 1$, *pour* $k' = k'_0, k'_0 + 1, k'_0 + 2, ...$ *tel que* $|s'| \simeq |t"| \simeq |t'|$. *Noter cet entier premier par* $s' = 2k's" + 1$.

(b) *Sélectionner un entier* i_0 . *Trouver le premier entier premier dans la suite* $2it" + 1$, *pour* $i = i_0, i_0 + 1, i_0 + 2, ...$ *Noter cet entier premier par* $d = 2it" + 1$.

(c) *sélectionner un entier* i'_0. *Trouver le premier entier premier dans la suite* $2i't' + 1$, *pour* $i' = i'_0, i'_0 + 1, i'_0 + 2, ...$ *Noter cet entier premier par* $d' = 2i't' + 1$. *Répéter (b) jusqu'à ce que* $d \neq d'$ *et* dd' *soient copremiers avec* s' ; *et optionnellement, jusqu'à ce que* $|d| = |d'|$.

(3) *calculer* $p_0 = 2[s'^{(d-1)(d'-1)-1} \mod (dd')]s' - 1$.

(4) *sélectionner un entier* j_0 . *Trouver le premier entier premier dans la suite* $p_0 + 2jdd's'$, *pour* $j = j_0, j_0 + 1, j_0 + 2, ...$ *Noter cet entier premier par* $p = p_0 + 2jdd's'$.

(5) *retourner* $((d, p))$.

Preuve : Montrons que le nombre premier p retourné par le nouvel algorithme modifié de Gordon est effectivement un entier premier très grand tel que $p - 1$ est divisible par dd' où d et d' sont des entiers premiers et, $d - 1$ et $d' - 1$ sont des entiers non friables.

Comme dd' et s' sont copremiers, alors, par le Théorème d'Euler, nous avons $s'^{(d-1)(d'-1)} = 1 \mod (dd')$.

D'où, il est facile de voir que $p_0 = 1 \mod (dd')$ et $p_0 = -1(\mod s')$.

Finalement, nous avons :

(i) $p - 1 = p_0 + 2jdd's' - 1 = 0 \mod (dd')$, et d'où $p - 1$ a le facteur dd' ;

(ii) $p + 1 = p_0 + 2jdd's' + 1 = 0 \mod (s)$, et d'où $p + 1$ a le facteur premier s' ;

(iii) $d - 1 = 2it" = 0 \mod (t")$, et d'où $d - 1$ a le facteur premier $t"$;

(iv) $d' - 1 = 2i't' = 0 \mod (t')$, et d'où $d' - 1$ a le facteur premier t' ;

(v) $s' - 1 = 2k's" = 0 \mod (s")$, et d'où $s' - 1$ a le facteur premier $s"$.

D'où, suivant la définition d'un nombre premier très grand (voir : Menezes et *al.* [30]), nous avons le résultat désiré.

Remarque 4.2.5. *On peut décrire ce schéma dans* $\frac{Z}{nZ}$ *avec* n *un module RSA ou sur les courbes elliptiques.*

4.3 Quelques schémas modifiés de signature Meta-El Gamal

Dans cette section, nous décrivons une signature numérique avec appendice sur un message binaire de taille arbitraire qui généralise "les Schémas de Signature Meta-El Gamal" de Horster et *al.* [21]. Supposons que tout élément $h' \in G$ peut être représenté en binaire c'est à dire, il existe une fonction injective $h : G \to \{0, 1\}^*$.

Supposons aussi qu'il existe une fonction de hachage cryptographique $H : \{0, 1\}^* \to \frac{\mathbb{Z}}{n\mathbb{Z}}$ où $n = \#G$.

Comme dans notre clé privée nous avons trois paramètres secrets à savoir (r, s, t) tandis que dans le schéma classique du DSA ou celui d'El Gamal il existe un unique paramètre, alors on peut concevoir au moins deux versions (ou familles) pour les schémas de signature avec notre système que ceux "des Schémas de Signature Meta-El Gamal" de Horster et *al.*. Nous allons étudier ces deux versions dans la suite.

4.3.1 Première version des schémas modifiés de DSA/Signature Meta-El Gamal

Signature et vérification

La seule différence avec l'algorithme de génération de clé pour le chiffrement précédent est le fait que, nous avons besoin parfois que **pgcd(r,n) = 1** et/ou **pgcd(t,n) = 1**.

NB : Voir ci-dessous dans la sous section 4.3.2 pour la seconde version de l'algorithme de génération de clé pour la signature

Algorithme 4.3.1. ***Signature***

Pour signer un message m, Bob devra faire ce qui suit.

(1) *calculer $H(h(m))$ qui appartient à $\frac{\mathbb{Z}}{n\mathbb{Z}}$;*

(2) *choisir un entier aléatoire $2 < \beta < n$, avec $pgcd(\beta, n) = 1$, β et $d - \beta$ suffisamment grands ;*

(3) *calculer $R = (g^s)^{r\beta}$ dans G, et $H(h(R))$, $(r\beta)^{-1} \mod n$ et $s^{-1}t \mod n$ qui appartiennent à $\frac{\mathbb{Z}}{n\mathbb{Z}}$;*

(4) *calculer $S = (r\beta)^{-1}\{H(h(m)) - s^{-1}tH(h(R))\} \mod n$; si $S = 0$ ou $S = 1$, retourner à l'étape 3 ;*

(5) *la signature de m est (R, S) ;*

Remarque 4.3.1. *R et r (respectivement : S et s) n'ont pas la même signification dans cette section.*

Algorithme 4.3.2. ***Vérification***

Pour vérifier une signature (R, S) de m avec la clé publique de Bob, Alice devra faire ce qui suit.

(1) *obtenir $((g^s, g^t), G)$ la clé publique de Bob et (R, S) la signature de Bob ;*

(2) *calculer $V_1 = (g^t)^{H(h(R))}R^S$ et $V_2 = (g^s)^{H(h(m))}$ dans G ;*

(3) *accepter la signature si et seulement si $V_1 = V_2$;*

Pourquoi la vérification marche ?

Comme $S = (r\beta)^{-1}\{H(h(m)) - s^{-1}tH(h(R))\} \mod n$ alors

$H(h(m)) = r\beta S + s^{-1}tH(h(R)) \mod n.$

$V_2 = (g^s)^{H(h(m))} = (g^s)^{[r\beta S + s^{-1}tH(h(R)) \mod n]} = (g^s)^{r\beta S}(g^s)^{s^{-1}tH(h(R))} = R^S(g^t)^{H(h(R))} = V_1$

Sécurité et performance

1. Sécurité

a) Paramètres aléatoires : Si le même β est utilisé dans les deux différentes signatures (R, S_1) et (R, S_2) avec $S_1 - S_2$ inversible, alors

$$r\beta = (S_1 - S_2)^{-1}[H(h(m_1)) - H(h(m_2))] \mod n \qquad (E1)$$

Comme :

$$S = (r\beta)^{-1}\{H(h(m)) - s^{-1}tH(h(R))\} \mod n \qquad (E2)$$

alors, si $H(h(R))$ est inversible, nous avons :

$$s^{-1}t = [H(h(R))]^{-1}\{H(h(m_1)) - r\beta S_1\} \mod n \qquad (E3)$$

Maintenant, l'attaquant, Charlie connaît $s^{-1}t \mod n$ et $r\beta \mod n$, mais $kd = rs + t$ d'où si $d = n$ alors $s^{-1}t \mod n = -r \mod d = d - r$. Ainsi Charlie connaît r. Mais, si d et n sont différents, il est difficile pour l'attaquant de calculer $r \mod n$ ce qui autorise avec $s^{-1}t \mod n$, à signer avec la clé privée de Bob. Toutefois, il est bon pour la sécurité d'utiliser une valeur aléatoire différente β pour chaque signature ! Nous proposons la suite dans (b), une modification telle que la vulnérabilité précédente ne marche pas même si le même paramètre aléatoire est utilisé pour deux signatures distinctes de deux messages.

b) Petite modification pour une meilleure sécurité : Nous remarquons que pour la signature R ne dépend pas du message m et notre équation fondamentale $kd = rs + t$ est non utilisée explicitement dans S ! Pour faire cela, nous proposons ce qui suit :

- choisir β tel que $r(\beta + Hh(m))$ soit inversible modulo n et calculer $(r(\beta + Hh(m)))^{-1} \mod n$,
- modifier R en calculant $R = (g^s)^{r(\beta+Hh(m))}$;
- modifier S en calculant :
  $S = (r(\beta + Hh(m)))^{-1}\{H(h(m)) - s^{-1}tH(h(R)) + r\} \mod n$;
- modifier V_1 en calculant $V_1 = (g^t)^{[1+H(h(R))]}R^S$;
- V_2 est non changé.

Avec cette modification, l'attaque connue pour la signature d'El Gamal pour des "signatures distinctes avec la même valeur aléatoire" ne marche pas.

c) Contrefaçon existentielle (Existential forgery) : Comme, dans notre schéma de signature, nous utilisons une fonction de hachage à savoir Hh, les deux méthodes d'attaques de contrefaçons existentielles décrites dans El Gamal [17] ou Pointcheval-Stern [37] ne marchent pas avec notre mécanisme de signature [où le calcul avec les paramètres et les valeurs du hachage sont effectués " mod n" au lieu de " mod d" tant que le calcul avec les éléments du groupe est effecté dans G].

d) Générateurs faibles (Weak generators) : La plupart des attaques connues basées sur les générateurs faibles peuvent être évitées par quelques techniques (voir Menezes et *al.*, [30], Bleichenbacher [6] et Suiyan [44]) etc.

e) Données publiques chiffrées (Public computing encrypted data) : Comme $R = (g^s)^{r\beta}$, g^s et g^t sont publiques, alors Charlie peut faire le calcul suivant :

$Rg^t = g^s)^{r\beta} g^t = g^{sr\beta+t} = g^{(sr\beta+t)} = g^{(kd\beta+(1-\beta)t)} = g^{(1-\beta)t}$.

Comme t est suffisamment grand et il est supposé que le logarithme discret est insoluble, alors il est difficile de calculer β même si le générateur est connu.

2. Performance . Relativement à la signature El Gamal, dans notre processus de signature, il existe trois opérations à savoir : une inversion s^{-1} et deux multiplications $r\beta$ et $s^{-1}t$. Remarquant que le calcul de s^{-1} peut être inclus dans le mécanisme de génération de clé, nous voyons que la seule différence pour le nombre d'opérations dans les deux systèmes, est les "deux multiplications". De plus, dans le but de réduire la taille de la clé privée pour cette signature, nous pouvons enregistrer seulement et garder secret $(r, s^{-1}t \mod n)$ dans l'algorithme de génération de clé au lieu de (r, s, t). Nous concluons que la performance caractéristique de ce processus de signature est similaire à celle de la signature classique d'El Gamal.

4.3.2 Seconde version des schémas modifiés de signature de Meta-El Gamal

Génération de clé

On peut décrire les signatures en utilisant la condition $pgcd(t, n) = 1$ au lieu de $pgcd(s, n) = 1$ et $pgcd(r, n) = 1$ dans l'algorithme de génération de clé pour la signature.

Signature

Prendre $R = (g^t)^\beta$ au lieu de $R = (g^s)^{r\beta}$ et calculer $S = (\beta)^{-1}\{H(h(m)) - t^{-1}sH(h(R))\} \mod n$ au lieu de $S = (r\beta)^{-1}\{H(h(m)) - s^{-1}tH(h(R))\}$ dans le mécanisme de signature ci-dessus. 4.3.1.

Vérification

La nouvelle vérification d'égalité est : $V_1 = R^S(g^s)^{H(h(R))} = V_2 = (g^t)^{H(h(m))}$ au lieu de $V_1 = R^S(g^t)^{H(h(R))} = V_2 = (g^s)^{H(h(m))}$, dans le processus de vérification ci-dessus 4.3.2.

Performance et sécurité

1) Sécurité :

(a) : La plupart des commentaires ci-dessus pour la sécurité dans 4.3.2 (pour la première version), s'appliquent à cette seconde version même si dans ce cas, seulement deux des trois paramètres de la clé privée sont utilisés. C'est à dire (s, t) est utilisé au lieu de (r, s, t) pour la seconde version du mécanisme de signature. Mais nous pouvons modifier le processus encore pour utiliser (r, s, t) pour cette seconde signature dans cette méthode suivante.

(b) Une nouvelle petite modification pour une meilleure sécurité : Nous remarquons que pour la signature, R ne dépend pas du message m. Pour faire cela, nous proposons ce qui suit :

- choisir β tel que $\beta + Hh(m)$ soit inversible modulo n et calculer $(\beta + Hh(m))^{-1} \mod n$, et $r^{-1} \mod n$;
- modifier R en calculant $R = (g^t)^{(\beta+Hh(m))}$;

- modifier S en calculant $S = (\beta + Hh(m))^{-1}\{H(h(m)) - t^{-1}sH(h(R)) + r^{-1}\} \mod n$;
- modifier V_1 en calculant $V_1 = g^{s[1+H(h(R))}R^S$;
- V_2 est non changé.

2) Performance : Cette seconde version est un peu plus efficace que la première version parce que elle utilise β au lieu de $r\beta$ et st^{-1} au lieu de $s^{-1}t$ dans le processus de signature.

4.3.3 Une Variante Modifiée des schémas de signature Meta-El Gamal

Première version de la variante modifiée des schémas de signature Meta-El Gamal

Toutes les variantes des schémas de signature Meta-El Gamal dans [21] (voir aussi Chap 11 de [30]), peuvent être généralisées avec nos schémas modifiés. La définition de R est inchangée (relativement au cas ci-dessus) c'est à dire $R = (g^s)^{r\beta}$, mais S est définie via l'équation de signature (similaire à celle de [21]) :

$$\textbf{Eq2 :} \quad u = s^{-1}tv + r\beta w$$

Il n'est pas difficile de vérifier que, avec Eq2, nous avons six possibilités pour les schémas de signature que nous listons dans le tableau suivant (voir 4.3.1 et 4.3.2).

	u	v	w	Vérification : $V_1 = V_2$
1	$H(h(m))$	$H(h(R))$	S	$R^S(g^t)^{H(h(R))} = (g^s)^{H(h(m))}$
2	$H(h(m))$	S	$H(h(R))$	$R^{H(h(R))}(g^t)^S = (g^s)^{H(h(m))}$
3	S	$H(h(R))$	$H(h(m))$	$R^{H(h(m))}(g^t)^{H(h(R))} = (g^s)^S$
4	S	$H(h(m))$	$H(h(R))$	$R^{H(h(R))}(g^t)^{H(h(m))} = (g^s)^S$
5	$H(h(R))$	S	$H(h(m))$	$R^{H(h(m)}(g^t)^S = (g^s)^{H(h(R))}$
6	$H(h(R))$	$H(h(m))$	S	$R^S(g^t)^{H(h(m))} = (g^s)^{H(h(R))}$

Seconde Version de la variante modifiée des schémas de signature Meta-El Gamal

On peut décrire les signatures en utilisant la condition $pgcd(t, n) = 1$ au lieu de $pgcd(r, n) = 1$ dans l'algorithme de génération clé ci-dessus.

D'où, en posant $R = (g^t)^\beta$, nous avons six possibilités de signature comme ci-dessus dans 4.3.3 et dans 4.3.2 avec l'équation de signature suivante :

$$\textbf{Eq3 :} \quad u = t^{-1}sv + \beta w$$

Par exemple si $u = H(h(m))$, $v = H(h(R))$ et $w = S$, l'équation de signature correspondante est $H(h(m)) = t^{-1}sH(h(R)) + \beta S$ et l'égalité de vérification est : $V_1 = R^S(g^s)^{H(h(R))} = V_2 = (g^t)^{H(h(m))}$

4.4 Résumé comparatif par rapport à El Gamal

– le message chiffré est deux fois plus long que le message d'origine comme El Gamal ;

- l'algorithme de génération de clé est plus lent que celui d'El Gamal mais pas beaucoup (négatif) ;
- la clé privée est petite (positif) ;
- le générateur g et son ordre d peuvent être privés (positif) ;
- certaines vulnérabilités dans la version "basic" (non standardisé) de El Gamal ne marchent pas dans notre algorithme (positif) ;
- le chiffrement a la même sécurité calculatoire que El Gamal (positif) ;
- notre schéma de signature peut être rédigé en 24 versions alors que celui de El Gamal en a 6 (positif) ;
- la signature DSS issue de El Gamal est sous Licence alors que celle-ci pourrait rester libre si elle est adoptée (positif) ;
- la signature et le déchiffrement peuvent être plus rapides que celui d'El Gamal car la clé privée est petite (positif), mais plus longue pour la signature (négatif).

Chapitre 5

Cryptanalyse et amélioration du Protocole d'échange de clés SDH-DSA-KE

Dans ce chapitre, nous avons fait une cryptanalyse du protocole nommé "Strong Diffie-Hellman-DSA Key Exchange (SDH-DSA-KE)" et après nous avons proposé le protocole "Strong Diffie-Hellman-Exponential-Schnnor Key Exchange (SDH-XS-KE)" qui est une amélioration pour l'efficience et la sécurité. Le protocole SDH-XS-KE est sûr contre les attaques "Session State Reveal (SSR) attack", l'indépendance des clés (Key independency attacks), "Unknown-key share (UKS) attacks" et "Key-Compromise Impersonation (KCI) attacks". De plus, SDH-XS-KE a la propriété appelée "Perfect Forward Secrecy (PFS)" et possède une étape de confirmation de clé. Cette nouvelle proposition est non vulnérable à "Disclosure to ephemeral or long-term Diffie-Hellman exponents". Nous décrivons notre protocole dans les groupes finis, et donc, ce protocole peut être implémenté sur les courbes elliptiques.

5.1 Introduction

Le protocole Diffie-Hellman (DH) [11] est le protocole d'échange de clé le plus populaire. Comme le protocole classique est vulnérable à une large catégories d'attaques, beaucoup de propositions ont été faites pour améliorer la sécurité du protocole DH (voir : [34], [28]). Mais, la plupart de ces propositions ont été cassées ou il a été démontré qu'elles souffrent de certaines défaillances.

Comme cela est dit dans [23], pour garantir l'authentification de l'échange de clé Diffie- Hellman [11], Arazi utilisait l'algorithme de signature DSA (Digital Signature Algorithm) [2]. Malheureusement le schéma de l'échange de clé intégré de [2] ne peut pas parer l'attaque dite "key independence" [34]. Harn et *al.* ont modifié le schéma de [11] pour parer l'attaque "key independence" [20]. Mais le schéma dans [20] ne peut pas parer l'attaque dite "forward secrecy" [34]. Phan a modifié le schéma de [20] pour parer l'attaque "forward secrecy" [36].

En 2007, dans le journal IEEE Communications letters (voir : [23]), Jeong et *al.* prouvent que le schéma précédent est non sûr contre l'attaque "session state reveal". Après, les auteurs ont proposé le papier dénommé "Strong Diffie-Hellman-DSA Key Exchange" (SDH-DSA-KE) où l'authentification mutuelle est effectuée par la signature DSA mais leur algorithme utilise 5 exposants et est vulnérable à certaines attaques.

Dans notre article, nous avons proposé une cryptanalyse de SDH-DSA-KE en montrant qu'il est non sûr contre les attaques KCI et est vulnérable à "Disclosure to ephemeral and long-term CDH exponents". Après, nous avons proposé un nouveau protocole d'échange nommé "Strong Diffie-Hellman-Exponential-Schnorr Key Exchange" (SDH-XS-KE) qui est une amélioration de SDH-DSA-KE en terme d'efficience et de sécurité. Notre protocole utilise 4 exposants et est sûr contre les attaques dites "Session State Reveal (SSR)", "Key independency", "Unknown-key share (UKS)" et "Key-Compromise Impersonation (KCI)". De plus, SDH-XS-KE a la propriété "Perfect Forward Secrecy (PFS)". Pour l'authentification mutuelle, au lieu de la signature DSA, nous avons utilisé le protocole Exponentiel de Schnorr modifié.

Notez que SDH-DSA-KE a été conçu uniquement dans $\mathbb{Z}/p\mathbb{Z}$ mais notre protocole est conçu pour un groupe (multiplicatif) fini arbitraire par conséquent notre protocole peut être implémenté sur les courbes elliptiques.

5.2 Préliminaires

5.2.1 Echange de clé Diffie-Hellman

Définition 5.2.1. *Un protocole cryptographique est une suite de règles déterminant l'ensemble des opérations cryptographiques nécessaires et leur séquence pour sécuriser une communication (une transaction, un échange de données, ..) entre plusieurs entités.*

Le protocole d'échange de clé Diffie-Hellman est développé en 1976 et publié dans l'article : *New directions in cryptography* [11]. Pour les cryptosystèmes symétriques, l'échange d'une clé cryptographique secrète est essentiel.

En effet, tout chiffrement d'une grande quantité de données doit être fait avec un chiffrement à clé secrète donc nécessite un échange de clé au préalable.

5.2.2 Problème du Logarithme Discret

Nous rappelons ici le Problème du Logarithme Discret pour harmoniser les notations de ce chapitre.

Le problème du Logarithme Discret est le suivant : on donne un groupe fini G d'ordre n et un sous-groupe cyclique $\langle g\rangle$ d'ordre premier q généré par g, si $y \overset{Rand}{\leftarrow} \langle g\rangle$, trouver l'entier x, $0 \leq x \leq q-1$, tel que $g^x = y$.

Le Problème Calculatoire de Diffie-Hellman (CDH) est le suivant : on donne un groupe fini G d'ordre n et un sous-groupe cyclique $\langle g\rangle$ d'ordre premier q généré par g, si $y = g^a \overset{Rand}{\leftarrow} \langle g\rangle$ et

$z = g^b \overset{Rand}{\leftarrow} \langle g \rangle$, trouver l'élément g^{ab}.

5.2.3 Protocole d'identification de Schnorr

Soit G un groupe multiplicatif et $\langle g \rangle$ un sous-groupe cyclique d'ordre premier q avec un générateur $g \in G$. La clé secrète sk est un entier x dans $]1, q[$. Prendre $y = g^x$, la clé publique pk est (G, g, y).

Dans ce protocole le "prouveur" est $\mathcal{P}$ et le vérifieur est $\mathcal{V}$.

1. $\mathcal{P}$ choisit un aléa $v \overset{Rand}{\leftarrow}]1, q[$ et envoie $V = g^v$ à $\mathcal{V}$.
2. $\mathcal{V}$ choisit un aléa "challenge" $e \overset{Rand}{\leftarrow}]1, q[$ et envoie e à $\mathcal{P}$.
3. $\mathcal{P}$ calcule $s = v + xe \ (\mod q)$ et envoie s à $\mathcal{V}$.

$\mathcal{V}$ accepte si et seulement si $g^s = Vy^e$.

Il est bien de connu que ce protocole est une preuve à divulgation nulle de la connaissance de x (voir :[28]) pour un vérifieur honnête $\mathcal{V}$ (i.e., quelqu'un qui choisit uniformément un aléa e).

Protocole d'identification de Schnorr exponentiel.

Soit G un groupe multiplicatif et $\langle g \rangle$ un sous-groupe cyclique d'ordre premier q avec un générateur $g \in G$.

Dans ce protocole, le "prouveur" est $\mathcal{P}$ et le vérifieur est $\mathcal{V}$.

La clé secrète sk de $\mathcal{P}$ est un entier x dans $]1, q[$. Prendre $y = g^x$, la clé publique pk de $\mathcal{P}$ est (G, g, y).

1. $\mathcal{V}$ choisit un aléa $w \overset{Rand}{\leftarrow}]1, q[$ et envoie le "challenge" $W = g^w$ à $\mathcal{P}$.
2. $\mathcal{P}$ choisit un aléa $v \overset{Rand}{\leftarrow}]1, q[$ et envoie $V = g^v$ à $\mathcal{V}$.
3. $\mathcal{V}$ choisit un aléa "challenge" $e \overset{Rand}{\leftarrow}]1, q[$ et envoie e à $\mathcal{P}$.
4. $\mathcal{P}$ calcule $s = v + xe \ (\mod q)$ et envoie $S = W^s$ à $\mathcal{V}$.

$\mathcal{V}$ accepte si et seulement si $S = (Vy^e)^w$.

Il est bien connu que ce protocole est une preuve "d'habileté" de $\mathcal{V}$ à calculer $CDH(y, V)$ pour toute valeur $V \in G$. De plus, le protocole est à divulgation nulle contre un vérifieur $\mathcal{V}$ qui choisit e comme aléa (tant que V peut être choisi arbitrairement). (voir : [28])

5.2.4 Notions de Sécurité sur les protocoles d'échange de clé

Rappelons quelques notions de sécurité utilisées dans les protocoles d'échange de clé.

Pour ne pas trahir le sens exact des définitions, nous donnons les versions en anglais puis une traduction qui tente de capter le sens en français.

1. **Key Independency**. "This is a stronger notion of security and means that session keys are computationaly independent from each other".

C'est une notion forte de sécurité. "Key independency" requiert que toutes les clés de session soient indépendantes entre elles.

2. **Session State Reveal Attack**. "The protocols providing security against session state reveal attacks maintain the secrecy of session keys even when an adversary is able to obtain the random numbers used to make the session keys".

   Pour garantir la sécurité contre une attaque SSR, il faut assurer la sécurité d'une clé de session même si les nombres aléatoires utilisés dans sa génération sont révélés.

3. **Perfect Forward Secrecy (PFS)** : "a key-exchange protocol is said to have the PFS property if the leakage of the long-term key of a party does not compromise the security of session keys established by that party and erased from memory before the leakage occurred".

   Un protocole d'échange a la propriété PFS si la révélation de la clé secrète utilisée pour générer toutes les clés de session ne compromet pas la sécurité d'une clé de session déjà utilisée et effacée en mémoire.

4. **Resistance to Key-Compromise Impersonation (KCI) attacks**" it provides the assurance that sessions established by a party Alice while not being actively controlled by the attacker, remains secure even if her private key is learned by the attacker".

   La résistance à l'attaque KCI garantit que la sécurité d'une clé de session établie par une partie (Alice) même si l'attaquant connait la clé privée d'Alice, étant entendu que l'attaquant ne contrôle pas entièrement la session d'Alice.

5. **Le cas du protocole d'échange de clé Diffie-Hellman :**(voir : [28])

   Soit G un groupe fini, cyclique et g un générateur de G. Considérons une session $(id_A, id_B, V_A = g^{v_A}, V_B = g^{v_B})$ entre deux parties A et B où v_A et v_B sont des valeurs aléatoires choisies par A et B avec la paire de clés privée/publique suivante : (x_A, g^{x_A}) et (x_B, g^{x_B}) ; le calcul de la clé de session comporte les quatre valeurs secrètes x_A, x_B, v_A, v_B. Evidemment la divulgation de $\{x_A, v_A\}$, ou $\{x_B, v_B\}$, permet à l'attaquant de connaitre la clé de session.

   Pour la sécurité de la communication entre deux parties A et B, on doit prouver que la divulgation de l'une des paires de valeurs (excepté $\{x_A, v_A\}$ et $\{x_B, v_B\}$) dans l'ensemble $\{x_A, x_B, v_A, v_B\}$ est insuffisant pour l'attaquant de réussir une attaque. Ceci inclut les cas dans lesquels l'attaquant connait :

   - $\{x_A, x_B\}$ et essaie de calculer la clé de session d'une session passée : c'est la propriété PFS ;
   - $\{v_A, v_B\}$: ceci correspond à la sécurité de l'attaque "Session State Reveal" ;
   - $\{x_A, v_B\}$ ou $\{x_B, v_A\}$: ceci correspond à la sécurité des attaques KCI ;
   - $g^{x_A x_B}$ sans connaitre (x_A, x_B) : ceci correspond à la sécurité de la divulgation des exposants DH de long-term ;
   - $g^{v_A v_B}$, sans connaitre $\{v_A, v_B\}$: ceci correspond à la sécurité de la divulgation des exposants DH éphémères ;

Une preuve de sécurité d'un protocole d'échange de clé procéde généralement en prouvant que si un attaquant $\mathcal{A}$ est capable en temps polynomial de casser l'une des notions de sécurité précédentes,

on peut utiliser un algorithme de simulation $\mathcal{S}$ pour inverser en temps polynomial une fonction à sens unique connue ou résoudre en temps polynomial un problème difficile.

Soit $\mathcal{A}$ un attaquant qui est capable de casser l'une des notions de sécurité précédentes en un temps τ_A avec une probabilité de succès au moins ε_A, alors pour la preuve de sécurité, $\mathcal{S}$ doit pouvoir simuler l'environnement de $\mathcal{A}$ et résoudre le problème connu avec un temps $\tau_S \geq \tau_A$ et une probabilité de succès $\varepsilon_S \geq \varepsilon_A$.

Pour la qualité de la réduction dans la preuve de sécurité, il est requis d'avoir $\varepsilon_S \approx \varepsilon_A$ et $\tau_S \approx \tau_A + \text{polynom}(k)$ où k est un paramètre de sécurité et polynom(k) est un polynôme en k.

Comme déjà vu dans les rappels, pour une preuve dans le Modèle de "l'Oracle aléatoire", (voir [4] et[8]) :

- une fonction de hachage est utilisée comme une fonction aléatoire dans le processus de simulation (elle est l'oracle de hachage) ;
- le seul moyen de calculer la fonction de hachage est d'invoquer l'oracle de hachage ;
- L'algorithme de simulation $\mathcal{S}$ doit simuler l'environnement de l'attaquant $\mathcal{A}$ avec une information publique seulement ;
- à la fin de la preuve, si l'attaquant $\mathcal{A}$ réussit dans quelques attaque (et par conséquent sort une clé de session partagée valide au moins avec l'une des deux parties : Alice ou Bob) alors $\mathcal{S}$ peut être en mesure de résoudre le problème difficile connu de la notion de sécurité.

5.3 Attaques sur l'échange de clé "Strong DH-DSA"

5.3.1 Protocole SDH-DSA-KE

Rappelons le procédé du protocole d'échange de clé "Strong DH-DSA" de [23].

Soient p, q deux nombres premiers suffisamment grands tel que q divise $p-1$ et $g \in \mathbb{Z}/p\mathbb{Z}$ un élément d'ordre q. Soit $H : \{0,1\}^* \to \mathbb{Z}/q\mathbb{Z}$ une fonction de hachage.

On suppose que Alice (respectivement : Bob) a une paire de clés privée/publique $(x_A, y_A = g^{x_A})$ $\big($respectivement : $(x_B, y_B = g^{x_B})$ $\big)$.

1. Alice génère $v_A \overset{Rand}{\longleftarrow}]1, q[$, calcule $m_A = g^{v_A} \mod p$ et envoie m_A à Bob.
2. – Bob génère $v_B \overset{Rand}{\longleftarrow}]1, q[$ et calcule $m_B = g^{v_B} \mod p$,
   – Bob calcule $DH1 = m_A^{v_B} \mod p = g^{v_A v_B} \mod p$, $DH2 = y_A^{x_B} \mod p = g^{x_A x_B} \mod p$, $r_B = m_B \mod p$
   – Bob signe $s_B = \big(v_B^{-1}(H(m_B||DH1||DH2) + x_B r_B)\big) \mod q$ avec la signature DSA.
   – Bob envoie (m_B, s_B) à Alice.
3. – Alice calcule $DH1 = m_B^{v_A} \mod p = g^{v_A v_B} \mod p$, $DH2 = y_B^{x_A} \mod p = g^{x_A x_B} \mod p$, $r_B = m_B \mod q$, $r_A = m_A \mod q$.
   – Alice vérifie DSA.ver$_{y_B}\big(m_B||DH1||DH2, r_B, s_B\big) \overset{?}{=} 1$
   – Alice calcule $K_{AB} = H(A||B||DH1||DH2)$ et $K_{BA} = H(B||A||DH1||DH2)$,

– Alice signe $s_A = \left(v_A^{-1}(H(m_A||DH1||DH2) + x_A r_A)\right) \mod q$ avec la signature DSA ;
– Alice envoie s_A à Bob.

4. – Bob calcule $r_A = m_A \mod q$.
– Bob vérifie si $\text{DSA.ver}_{y_A}\left(m_A||DH1||DH2, r_A, s_A\right) \stackrel{?}{=} 1$
– Bob calcule $K_{AB} = H(A||B||DH1||DH2)$ et $K_{BA} = H(B||A||DH1||DH2)$.

5.3.2 Cryptanalyse du protocole SDH-DSA-KE

Attaque KCI sur SDH-DSA-KE

Théorème 5.3.1. *SDH-DSA-KE est non sûr contre l'attaque Key-Compromise Impersonation (KCI).*

Preuve : Supposons que l'attaquant connait x_A et v_B, puisque $y_B = g^{x_B} \mod p$ et $m_A = g^{v_A} \mod p$ sont publiques alors l'attaquant peut facilement calculer $DH1 = m_A^{v_B} \mod p = g^{v_A v_B} \mod p$, $DH2 = y_B^{x_A} \mod p = g^{x_A x_B} \mod q$ et déduire la clé. Nous avons le même résultat si l'attaquant connait (x_B et v_A). D'où ce protocole est vulnérable à l'attaque Key-Compromise Impersonation (KCI).

□

Disclosure to ephemeral or long-term CDH exponents.
Dans le protocole SDH-DSA-KE, les clés sont calculées comme suit : $K_{AB} = \overline{H}_1(A||B||DH1||DH2)$ et $K_{BA} = \overline{H}_1(B||A||DH1||DH2)$. Donc, dans ce protocole la valeur $DH2 = g^{x_A x_B} \mod p$ sert comme clé partagée de long terme entre les parties Alice et Bob, et donc cette divulgation suffit pour se faire passer pour Alice à Bob et vice versa.

Nous verrons que pour notre amélioration, le dévoilement de $DH2 = g^{x_A x_B} \mod p$ ne permet pas l'usurpation d'Alice ou de Bob.

□

5.4 Nouvelle proposition : modèle de protocole SDH-XS-KE

Dans notre protocole l'authentification mutuelle est effectuée par le protocole d'exponentiation de Schnorr modifié où chaque partie utilise sa clé publique.

5.4.1 Protocole SDH-XS-KE

Soit G un groupe multiplicatif et $\langle g \rangle$ un sous-groupe d'ordre premier q avec un générateur $g \in G$. Soit $H : G \times G \times \overline{\mathcal{P}} \to \{0,1\}^l$ une fonction de hachage (où $l \geq 224$ et $\overline{\mathcal{P}}$ est l'ensemble de toutes les parties qui sont autorisées à participer au protocole). Soient $\overline{H} : \{0,1\}^l \times \overline{\mathcal{P}} \times \overline{\mathcal{P}} \times \{0,1\} \to \{0,1\}^l$ une fonction de hachage et $\mathbf{MAC}_K : G \times G \times \overline{\mathcal{P}} \to \{0,1\}^l$ une fonction de hachage avec clé, utilisée pour l'authentification Mac.

On suppose qu'Alice (respectivement : Bob) a une paire de clés privée/publique $(x_A, y_A = g^{x_A})$ $\left(\text{respectivement : } (x_B, y_B = g^{x_B})\right)$ où $x_A, x_B < q$ sont des entiers aléatoires.

Protocole

1. Alice (l'initiatrice) choisit un aléa de session secret $v_A < q$, calcule $V_A = g^{v_A}$, $\delta_{AB} = y_B^{v_A+x_A}$ et $h_{AB} = H\big(\delta_{AB}, V_A, id_A\big)$, détruit δ_{AB} et envoie (V_A, h_{AB}) à Bob ;
2. – Bob (le répondeur) vérifie si $V_A \neq 1$, calcule $\lambda_{BA} = (V_A y_A)^{x_B}$ et $H(\lambda_{BA}, V_A, id_A)$ et détruit λ_{BA} ; vérifie si $H(\lambda_{BA}, V_A, id_A) \neq h_{AB}$;
   – Si l'une des validations précédentes échoue alors Bob termine le protocole qui s'arrête avec echec ;
   – Bob choisit un aléa de session secret $v_B < q$, calcule $V_B = g^{v_B}$ et $K_{mac} = \overline{H}(g_{KBs}, id_A, id_B, 1)$ où $g_{KBs} = (V_A y_A)^{v_B+x_B}$.
   – Bob calcule $\delta_{BA} = y_A^{v_B+x_B}$, $h_{MAC_B} = \mathbf{MAC}_{K_{mac}}\big(\delta_{BA}, V_B, id_B\big)$, détruit δ_{BA} et envoie (V_B, h_{MAC_B}) à Alice ;
3. – Alice vérifie si $V_B \neq 1$, calcule $K_{mac} = \overline{H}(g_{KAs}, id_A, id_B, 1)$ où $g_{KAs} = (V_B y_B)^{v_A+x_A}$.
   – Alice calcule $\lambda_{AB} = (V_B y_B)^{x_A}$ et $h'_{MAC_B} = \mathbf{MAC}_{K_{mac}}\big(\lambda_{AB}, V_B, id_B\big)$, et détruit λ_{AB} ; vérifie si $h'_{MAC_B} \neq h_{MAC_B}$,
   – Si l'une des validations précédentes échoue alors Alice termine le protocole qui s'arrête avec échec,
   – Alice calcule $h_{MAC_A} = \mathbf{MAC}_{K_{mac}}\big(g_{KAs}, V_A, id_A\big)$, et l'envoie à Bob.
   – Alice calcule et garde $K_{As} = \overline{H}(g_{KAs}, id_A, id_B, 0)$ comme sa clé de session courante.
4. – Bob calcule $h'_{MAC_A} = \mathbf{MAC}_{K_{mac}}\big(g_{KBs}, V_A, id_A\big)$, et vérifie si $h'_{MAC_A} \neq h_{MAC_A}$.
   – Si la validation échoue alors Bob termine le protocole qui s'arrête avec échec, sinon, Bob calcule et garde la clé $K_{Bs} = \overline{H}(g_{KBs}, id_A, id_B, 0)$ comme sa clé de session courante.

5.4.2 Sécurité et Performance de SDH-XS-KE

Dans notre protocole, l'authentification mutuelle est effectuée par le protocole Exponentiel de Schnorr modifié où chaque partie utilise sa clé publique. Ce protocole comporte une étape de confirmation de clé. Notez que l'exactitude du protocole dépend de l'honnêteté des parties établissant une session. Ils doivent choisir, calculer et garder correctement toutes les valeurs intervenant dans le protocole et aussi ils doivent suivre les étapes dans le bon chemin. Ainsi, dans la suite, on suppose que les parties sont honnêtes.

Performance

Notre protocole utilise 4 exposants seulement et 4 échanges avec une étape de confirmation de clé, ainsi il est plus rapide que SDH-DSA-KE qui utilise 5 exposants (3 exposants dans la génération des clés et 2 exposants dans le processus de vérification DSA) et 4 échanges.

Sécurité

Nous avons prouvé précédemment que SDH-DSA-KE est vulnérable à l'attaque KCI. Nous verrons dans la suite que notre protocole est sûr si CDH(V_A, V_B), CDH(y_A, y_B), CDH(V_A, y_B), CDH(V_B, y_A) sont calculatoirement difficiles et les fonctions de hachage sont robustes.

Dans les résultats suivants, les preuves qui sont effectuées dans le **modèle de l'oracle aléatoire** utilisent la simulation suivante.

Le simulateur $\mathcal{S}$ va simuler les deux fonctions de hachage $H, \overline{H}$ et sauvegarde dans deux bases de données toutes les paires $(t, H(t)) \in Hist[H]$ et $(z, \overline{H}(z)) \in Hist[\overline{H}]$ pour chaque requête de l'attaquant : soient t à H et z à $\overline{H}$. De plus, quand l'attaquant sollicite une des oracles de hachage (soit t à H), alors le simulateur $\mathcal{S}$ vérifie dans $Hist[H]$ si t était sollicité dans le passé à H. Si $H(t)$ est déjà défini comme $h_{H(t)}$ dans $Hist[H]$, $\mathcal{S}$ retourne $h_{H(t)}$ à $\mathcal{A}$, sinon $\mathcal{S}$ prend $h_{H(t)}$ une valeur aléatoire, retourne $h_{H(t)}$ comme hache de t à $\mathcal{A}$ et sauvegarde $(t, h_{H(t)})$ dans $Hist[H]$.

En outre, il est facile de voir que la réduction est optimale dans les preuves de sécurité dans le modèle de l'oracle aléatoire, avec $\varepsilon_S \approx \varepsilon_A$ et $\tau_S \approx \tau_A + \text{polynom}(k)$, car les opérations que nous utilisons ne modifient pas les probabilités en jeu.

Théorème 5.4.1. *Le protocole d'échange de clé SDH-XS-KE possède la propriété Perfect Forward Secrecy (PFS).*

Preuve : Le protocole a une étape de confirmation de clé alors l'attaquant ne peut pas être actif quand la clé de session est en train d'être construite par les deux parties. Le seul moyen pour l'attaquant, est d'essayer de calculer directement la clé de session en supposant qu'il connait les clés secrètes à long-terme (à savoir x_A, x_B) des deux parties.

Si l'attaquant sort une clé de session valide K, alors cette clé est calculée via $K = K_{Bs} = \overline{H}(g_{KAs}, id_A, id_B, 0)$ ou $K = K_{As} = \overline{H}(g_{KBs}, id_A, id_B, 0)$. où $g_{KBs} = g_{KAs} = g^{(v_A+x_A)(v_B+x_B)}$. L'attaquant $\mathcal{A}$ doit sortir à un moment donné dans ses tests la valeur g_{KBs} ou g_{KAs} et ainsi le simulateur peut trouver cette valeur dans la base de données d'oracle de hachage. Or $g_{KBs} = g_{KAs} = g^{v_A v_B} g^{x_A x_B} V_A^{x_B} V_B^{x_A}$ et le simulateur connaît x_A et x_B via l'attaquant alors il peut calculer $g^{x_A x_B} V_A^{x_B} V_B^{x_A}$. Donc le simulateur peut calculer $g^{(v_A v_B)} = CDH(V_A, V_B)$ avec la même probabilité de succès que l'attaquant et un temps similaire. Ainsi, si $CDH(V_A, V_B)$ est difficile, le protocole SDH-XS-KE possède la propriété PFS. □

Théorème 5.4.2. *Le protocole d'échange de clé SDH-XS-KE est sûr contre les attaques "Key-Compromise Impersonation (KCI) attacks" et "unknown-key share (UKS) attacks".*

Preuve :

1) Sécurité contre l'attaque "Key-Compromise Impersonation (KCI)" : Le protocole utilise une authentification mutuelle. Alice calcule $\delta_{AB} = y_B^{v_A+x_A}$ et envoie $h_{AB} = H\big(\delta_{AB}, V_A, id_A\big)$ à Bob et détruit δ_{AB} ; Bob calcule $\delta_{BA} = y_A^{v_B+x_B}$ et envoie $h_{MAC_B} = \mathbf{MAC}_{K_{mac}}\big(\delta_{BA}, V_B, id_B\big)$ à Alice et détruit δ_{BA}. D'où l'authentification de l'attaquant échoue s'il est actif et ne connaît pas simultanément v_A et x_A ou v_B et x_B.

Comme la preuve précédente, si l'attaquant sort une clé de session valide K, alors cette clé est calculée via $K = K_{Bs} = \overline{H}(g_{KAs}, id_A, id_B, 0)$ ou $K = K_{As} = \overline{H}(g_{KBs}, id_A, id_B, 0)$. où $g_{KBs} = g_{KAs} = g^{(v_A+x_A)(v_B+x_B)}$. L'attaquant $\mathcal{A}$ doit sortir à un moment donné dans ses tests la valeur $g_{KBs} = g_{KAs}$ et ainsi le simulateur peut trouver cette valeur dans la base de données de l'oracle de

hachage. Or $g_{KBs} = g_{KAs} = V_A^{v_B} y_B^{x_A} V_B^{x_A} V_A^{x_B}$ et le simulateur $\mathcal{S}$ connaît la clé secrète à long-terme de Alice (à savoir x_A) et la valeur aléatoire de session de Bob (à savoir v_B) alors $\mathcal{S}$ peut calculer $V_A^{v_B} y_B^{x_A} V_B^{x_A}$. D'où $\mathcal{S}$ peut calculer $V_A^{x_B} = CDH(V_A, y_B)$ avec la même probabilité de succès que l'attaquant et un temps similaire. Ainsi, si $CDH(V_A, y_B)$ est difficile, alors le protocole SDH-XS-KE est sûr contre l'attaque "KCI attacks".

2) Sécurité contre l'attaque "unknown-key share (UKS)" : Dans le processus d'authentification, Alice calcule $\delta_{AB} = y_B^{v_A + x_A}$ et envoie $h_{AB} = H\big(\delta_{AB}, V_A, id_A\big)$ à Bob et détruit δ_{AB} ; Bob calcule $\delta_{BA} = y_A^{v_B + x_B}$ et envoie $h_{MAC_B} = \mathbf{MAC}_{K_{mac}}\big(\delta_{BA}, V_B, id_B\big)$ à Alice et détruit δ_{BA}. D'où les clés publiques et les identités des deux parties (id_A, id_B) sont hachées. Et ceci est la méthode habituelle pour se protéger des attaques UKS. □

Théorème 5.4.3. *Le protocole d'échange de clé SDH-XS-KE est sûr contre l'attaque "Session State Reveal (SSR) attacks".*

Preuve : Comme la preuve précédente, si l'attaquant sort une clé de session valide K, alors cette clé est calculée via $K = K_{Bs} = \overline{H}(g_{KAs}, id_A, id_B, 0)$ ou $K = K_{As} = \overline{H}(g_{KBs}, id_A, id_B, 0)$. où $g_{KBs} = g_{KAs} = g^{(v_A + x_A)(v_B + x_B)}$. L'attaquant $\mathcal{A}$ doit sortir à un moment donné dans ses calculs la valeur g_{KBs} or g_{KAs} et donc le simulateur peut trouver cette valeur dans la base de données de l'oracle de hachage. Comme la clé de session est $K_{Bs} = \overline{H}(g_{KBs}, id_A, id_B, 0)$ où $g_{KBs} = g^{v_A v_B} g^{x_A x_B} y_A^{v_B} y_B^{v_A}$ et le simulateur $\mathcal{S}$ connaît v_A et v_B via l'attaquant $\mathcal{A}$, alors S peut calculer $g^{v_A v_B} y_A^{v_B} y_B^{v_A}$. D'où $\mathcal{S}$ peut calculer $g^{x_A x_B} = CDH(y_A, y_B)$ avec la même probabilité de succès que l'attaquant et un temps similaire. Donc, si $CDH(y_A, y_B)$ est difficile, le protocole SDH-XS-KE est sûr contre l'attaque "SSR attacks". □

Théorème 5.4.4. *Le protocole d'échange de clé SDH-XS-KE possède la propriété key independency.*

Preuve : Notez qu'une preuve rigoureuse pour l'attaque "Key Independency (KI)" est difficile à établir pour tous les protocoles d'échange de clé. Habituellement, la méthode des designers de protocoles d'échange de clés, est de fournir des arguments qui expliquent pourquoi leur protocole possède la propriété "Key Independency". Nos explications sont les suivantes.

Comme dans notre protocole, la clé de session est $K_{Bs} = \overline{H}(g_{KBs}, id_A, id_B, 0)$ où $g_{KBs} = g^{(v_A + x_A)(v_B + x_B)}$.

Alors la propriété "key independency" est garantie par :

- les propriétés des fonctions de hachage (les sorties des fonctions de hachage sont mutuellement indépendantes),
- l'usage des identités id_A et id_B des deux parties,
- et les valeurs aléatoires de session v_A and v_B.

□

Théorème 5.4.5. *Le protocole d'échange de clé SDH-XS-KE est sûr contre les attaques basées sur "disclosure to ephemeral et long-term CDH exponents" c'est à dire la divulgation des clés secrètes de long-terme ou de court-terme.*

Preuve : Si l'attaquant sort une clé de session valide K, alors cette clé est calculée via $K = K_{Bs} = \overline{H}(g_{KAs}, id_A, id_B, 0)$ ou $K = K_{As} = \overline{H}(g_{KBs}, id_A, id_B, 0)$. où $g_{KBs} = g_{KAs} = g^{(v_A+x_A)(v_B+x_B)}$. L'attaquant $\mathcal{A}$ doit sortir à un moment donné dans ses calculs la valeur g_{KBs} ou g_{KAs} et donc le simulateur peut trouver cette valeur dans la base de données de l'oracle de hachage.

Comme la clé de session est $K_{Bs} = \overline{H}(g_{KBs}, id_A, id_B, 0)$ où $g_{KBs} = g^{v_A v_B} g^{x_A x_B} y_A^{v_B} y_B^{v_A}$ et le simulateur $\mathcal{S}$ connaît $g^{v_A v_B}$ et $g^{x_A x_B}$ via l'attaquant $\mathcal{A}$, alors il peut calculer $g^{v_A v_B} g^{x_A x_B}$. D'où $\mathcal{S}$ peut calculer $y_A^{v_B} y_B^{v_A} = CDH(y_A, V_B)CDH(V_A, y_B)$ avec la même probabilité de succès que l'attaquant. Donc, si $CDH(y_A, V_B)$ et $CDH(V_A, y_B)$ sont difficiles, alors le protocole SDH-XS-KE est sûr contre les attaques de ce théorème. □

5.4.3 Tableau comparatif des protocoles HMQV, SDH-XS-KE et SDH-DSA-KE

	HMQV (Crypto 2005)	SDH-XS-KE (ici)	SDH-DSA-KE
Nombre d'échanges par partie	4	4	4
Nombre d'exponentiations	2,5	4	5
Preuve de sécurité	oui	oui	non
State reveal attack	oui	oui	oui ?
Perfect forward secrecy	non (oui pour la version faible)	oui	oui ?
KCI attack	oui	oui	non
Key independency	oui	oui	oui
Disclosure ephemerale et long-term, CDH exponents	oui	oui	non
Key confirmation step	oui (après modification et dans ce cas on a 5 échanges)	oui	non

Conclusion

Nous avons décrit avec succès dans ce chapitre une cryptanalyse sur le protocole SDH-DSA-KE et nous avons proposé un nouveau modèle de protocole d'échange de clé appelé SDH-XS-KE qui résiste à toutes les attaques connues sur les protocoles d'échange de clé.

Noter toutefois que notre protocole n'est pas aussi efficient que HMQV [28]. Notre protocole peut être implémenté sur les courbes elliptiques.

Chapitre 6

Etude de nouvelles formes de courbes elliptiques binaires

Article publié :

" Demba Sow, Djiby Sow : *"New Model of Binary Elliptic Curve"*. **Journal of Mathematics Research, Vol. 4, No. 6, December 2012**,

(http ://www.ccsenet.org/journal/index.php/jmr/article/view/22368). DOI : 10.5539/jmr.v4n6p34

"

Dans ce chapitre, nous proposons une nouvelle courbe elliptique binaire de la forme

$$a[x^2 + y^2 + xy + 1] + (a + b)[x^2y + y^2x] = 0 \qquad (\mathcal{E})$$

Si $m \geq 5$, nous prouvons que toute courbe elliptique ordinaire $y^2 + xy = x^3 + \alpha x^2 + \beta,\ \beta \neq 0$ sur $\mathbb{F}_{2^m}$, est birationnellement équivalente sur $\mathbb{F}_{2^m}$ à notre courbe $(\mathcal{E})$.

Nous présentons aussi, dans ce chapitre, les formules de la loi de groupe associée à $(\mathcal{E})$.

6.1 Introduction

Actuellement les courbes elliptiques sont largement étudiées et utilisées [3, 42]. Par exemple elles sont introduites en 1985 dans la cryptographie par Neal Koblitz [25, 26] et Victor Miller [32].

Rappelons qu'une courbe elliptique sur un corps $\mathbb{K}$ est une courbe projective non singulière de genre 1 avec au moins un point $\mathbb{K}$-rational. Plus explicitement, quand $\mathbb{K} = \mathbb{F}_2^m$, une courbe elliptique (non-supersingulière) peut être écrite comme une équation de Weierstrass classique

$$E_{\mathbb{K}} : y^2 + xy = x^3 + \alpha x^2 + \beta \qquad (\beta \neq 0).$$

avec un point à l'infini $O = (0 : 1 : 0)$.

Il y a deux types de corps finis utilisés pour l'implémentation des cryptosystèmes des courbes elliptiques [43] : les corps dont la charactéristique est un grand nombre premier et les corps de charactéristique 2 (appelé corps binaire).

Récemment, plusieurs articles sont écrits concernant les courbes elliptiques binaires telles que les courbes binaires d'Edwards (Binary Edwards curves) [5] et les courbes binaires de Huff (Binary Huff curves) [10].

Dans ce chapitre, nous introduisons une nouvelle courbe elliptique binaire.

Dans la section 1, nous introduisons une nouvelle courbe binaire et prouvons qu'elle est une variété projective.

Dans la section 2, nous étudions l'universalité du modèle et expliquons comment effectuer l'addition via une équivalence birationnelle.

Définitions : Notion de trace.

La trace est une fonction définie sur toute extension de corps fini. Dans cette section nous décrivons ses propriétés basiques.

Rappelons qu'une tansformation linéaire T d'un espace vectoriel V de dimension finie a un déterminant et une trace, qui sont le déterminant et la trace (somme des éléments de la diagonale) de la matrice de T dans une quelconque base de V. Si

$$c(X) = det(T - XI) = (-1)^n X^n + (-1)^{n-1} c_{n-1} X^{n-1} + \cdots + c_0$$

est le polynôme charatéristique de T, alors le déterminant de T est c_0 et sa trace est c_{n-1} . En particulier, le déterminant et la trace d'une matrice de T dans une base de V ne dépendent pas du choix d'une base.

Une extension finie E d'un corps K est un espace vectoriel de dimension finie sur K, et une multiplication par $\alpha \in E$ est une transformation linéaire $\gamma \longmapsto \alpha\gamma$ de E.

Définition 6.1.1. *Soit E une extension finie d'un corps K. La trace $Tr_K^E(\alpha)$ de $\alpha \in E$ sur K est la trace de la transformation linéaire $T_\alpha : \gamma \longmapsto \alpha\gamma$ de E.*

Propriété 6.1.1. *Soient $\alpha, \beta \in E = \mathbb{F}_q$ et $K = \mathbb{F}_p$ où $q = p^r$ avec p premier.*

1. *$Trace(\alpha) \in \mathbb{Z}/p\mathbb{Z}$;*
2. *$Trace(\alpha^p) = \alpha$;*
3. *Il existe $\gamma \in \mathbb{F}_{p^n}$, avec $Trace(\gamma) \neq 0$;*
4. *Si $a \in \mathbb{Z}/p\mathbb{Z}$, alors $Trace(a) = na$;*
5. *Si $a \in \mathbb{Z}/p\mathbb{Z}$, alors $Trace(a\alpha) = a\,Trace(\alpha)$;*
6. *$Trace(\alpha + \beta) = Trace(\alpha) + Trace(\beta)$;*
7. *Le polynôme $x^p - x - \alpha \in \mathbb{F}_q[x]$ est*
   a) soit irréductible ;
   b) ou un produit de facteurs de degré 1.
8. *Le polynôme $x^p - x - \alpha \in \mathbb{F}_q[x]$ est produit de facteurs de degré 1 si et seulement si $Trace(\alpha) = 0$.*

Corollaire 6.1.1. ***Trace d'une fonction pour les corps binaires ($p = 2$).***

Soient α $\beta \in \mathbb{F}_{2^r}$

1. *Trace*$(\alpha^2) = \alpha$;

2. *L'équation* $x^2+ux+v = 0$ *avec* $u,\ v \in \mathbb{F}_{2^r},\ u \neq 0$ *a une solution si et seulement si Trace*$(\frac{v}{u^2}) = 0$. *De plus, pour une solution* x_0 *l'autre est* $x_0 + u$.

6.2 Une nouvelle courbe elliptique binaire

Dans la suite, nous introduisons une nouvelle courbe et étudions ses propriétés.

Définition 6.2.1. ***(Nouvelle courbe binaire)***. *Soit* $\mathbb{K}$ *un corps de charactéristique* 2. *Soit* $(a,\ b)$ *un élément de* $\mathbb{K}^2$ *avec* $ab(a+b) \neq 0$. *La nouvelle courbe elliptique binaire avec les coefficients* a *et* b *est la courbe affine donnée par l'équation :*

$$\mathcal{E} : a[x^2 + y^2 + xy + 1] + (a + b)[x^2y + y^2x] = 0.$$

6.2.1 Variétés

Proposition 6.2.1. *La courbe* $a[x^2+y^2+xy+1]+(a+b)[x^2y+y^2x] = 0$ *avec* $ab(a+b) \neq 0$ *définie sur* $\mathbb{K}$ *est absolument irréductible dans* $\mathbb{K}$ *(donc est irréductible dans la clôture algébrique* $\overline{\mathbb{K}}$*).*

Preuve : Prenons $H(x, y) = [a + (a + b)x]y^2 + [ax + (a + b)x^2]y + a(x^2 + 1)$ dans $\mathbb{K}$. Supposons que H est réductible dans la clôture algébrique de $\mathbb{K}$ *i.e.* il existe quatre fonctions non toutes nulles f , f' , g *et* g' telles que $H(x, y) = [f(x) + g(x)y][f'(x) + g'(x)y] = f(x)f'(x) + (f(x)g'(x) + g(x)f'(x))y + g(x)g'(x)y^2$

par identification : $\begin{cases} f(x)f'(x) = a(x^2 + 1), & (1)\,; \\ g(x)g'(x) = (a + b)x + a, & (2)\,; \\ f(x)g'(x) + g(x)f'(x) = ax + (a + b)x^2, & (3). \end{cases}$

– $\underline{1^{er}\ \textbf{cas :}}$ $f = cste$ alors $(1) \Longrightarrow f' = \dfrac{a(x^2 + 1)}{f}$, $(3) \Longrightarrow g = cste$ et $(2) \Longrightarrow g' = \dfrac{a + (a + b)x}{g}$.

Dans (3) nous avons $fg' + gf' = a\dfrac{g}{f}x^2 + (a + b)\dfrac{f}{g}x + a(\dfrac{f}{g} + \dfrac{g}{f})$;

par identification $\begin{cases} a + b = a\dfrac{g}{f}, & (1')\,; \\ a = (a + b)\dfrac{f}{g}, & (2')\,; \\ a(\dfrac{f}{g} + \dfrac{g}{f}) = 0, & (3'). \end{cases} \Longleftrightarrow \begin{cases} \dfrac{g}{f} = \dfrac{a+b}{a}, & (1")\,; \\ \dfrac{f}{g} = \dfrac{a}{a+b}, & (2")\,; \\ a(\dfrac{a+b}{a} + \dfrac{a}{a+b}) = 0, & (3"). \end{cases}$.

$(3'')\dfrac{b^2}{a+b} = 0 \Longleftrightarrow b = 0$ impossible car $b \neq 0$

– $\underline{2^{me}\ \textbf{cas :}}$ $f' = cste$ alors $(1) \Longleftrightarrow f = \dfrac{a(x^2 + 1)}{f'}$, $(3) \Longleftrightarrow g' = cste$ et $(2) \Longleftrightarrow g = \dfrac{a + (a + b)x}{g'}$. Dans (3) nous avons $fg' + gf' = a\dfrac{g'}{f'}x^2 + (a + b)\dfrac{f'}{g'}x + a(\dfrac{f'}{g'} + \dfrac{g'}{f'})$;

$$\text{par identification} \begin{cases} a+b = a\dfrac{g'}{f'}, & (1')\,; \\ a = (a+b)\dfrac{f'}{g'}, & (2')\,; \\ a(\dfrac{f'}{g'} + \dfrac{g'}{f'}) = 0, & (3'). \end{cases} \iff \begin{cases} \dfrac{g'}{f'} = \dfrac{a+b}{a}, & (1")\,; \\ \dfrac{f'}{g'} = \dfrac{a}{a+b}, & (2")\,; \\ a(\dfrac{a+b}{a} + \dfrac{a}{a+b}) = 0, & (3"). \end{cases}$$

$(3'') \iff \dfrac{b^2}{a+b} = 0 \iff b = 0$ impossible car $b \neq 0$

– $\underline{3^{me}\ \textbf{cas :}}$ $\deg f = \deg f' = 1$ alors il existe a_1, *et* a_2 tels que $f(x) = a_1(x+1)$ et $f'(x) = a_2(x+1)$. L'équation (2) implique que $g = cste$ ou $g' = cste$. Supposons $g = cste$ alors $g' = \dfrac{a+(a+b)x}{g}$. L'équation (3) implique que $fg' + gf' = a_1(x+1)\dfrac{[(a+b)x+a]}{g} + ga_2(x+1) = x[(a+b)x+a]$ si $x = 1$ alors $(a+b)+a = 0$ impossible car $b \neq 0$.

□

6.2.2 Variétés non singulières

Théorème 6.2.1. ***(Non-singularité)****. Toute courbe elliptique binaire définie sur* $\mathbb{K}$ *par* $a[x^2+y^2+xy+1]+(a+b)[x^2y+y^2x] = 0$ *est non-singulière.*

Preuve : On a une variété non singulière si le système suivant n'admet pas de solutions :

$$\begin{cases} H(x,y) = y^2[a+(a+b)x] + y[ax+(a+b)x^2] + a(x^2+1) = 0, & (1)\,; \\ \frac{\partial H}{\partial x} = (a+b)y^2 + ay = 0, & (2)\,; \\ \frac{\partial H}{\partial y} = (a+b)x^2 + ax = 0, & (3). \end{cases}$$

L'équation (2) implique que $y = 0$ ou $y = \dfrac{a}{a+b}$ et l'équation (3) implique que $x = 0$ ou $x = \dfrac{a}{a+b}$.

Si $x = \dfrac{a}{a+b}$, dans (1) nous avons $a(\dfrac{a^2}{a^2+b^2}+1) = 0 \iff ab^2 = 0 \iff a = 0\ ou\ b = 0$, impossible car $ab \neq 0$. D'où H est non-singulière.

□

6.2.3 Forme projective

Equation homogène

Si nous prenons $x = \dfrac{X}{Z}$ et $y = \dfrac{Y}{Z}$, nous obtenons la forme projective de la courbe $\mathcal{E}$ qui correspond à l'équation homogène suivante :

$$a[X^2Z + Y^2Z + XYZ + Z^3] + (a+b)[X^2Y + Y^2X] = 0.$$

Points à l'infini

$Z = 0$ implique que $(a+b)[X^2Y + Y^2X] = 0$ si et seulement si. $X = 0$ ou $Y = 0$ ou $X = Y$.

– $X = 0$, $(X : Y : 0) = (0 : Y : 0) = (0 : 1 : 0)$;

– $Y = 0$, $(X : Y : 0) = (X : 0 : 0) = (1 : 0 : 0)$;

– $X = Y$, $(X : Y : 0) = (X : X : 0) = (1 : 1 : 0)$.

Ainsi, nous avons trois points à l'infini.

Singularité des points à l'infini

– $(1 : 0 : 0)$, $X = 1$, nous avons l'équation suivante :
$T(Z, Y) = a[Z + Y^2Z + YZ + Z^3] + (a + b)[Y + Y^2]$.
$\frac{\partial T}{\partial Y} = aZ + a + b$, $\frac{\partial T}{\partial Y}(0, 0) = a + b \neq 0$ d'où le point $(1 : 0 : 0)$ est un point infini non-singulier.

– $(0 : 1 : 0)$, $Y = 1$, nous avons l'équation suivante :
$T(X, Z) = a[X^2Z + Z + XZ + Z^3] + (a + b)[X^2 + X]$.
$\frac{\partial T}{\partial X} = aZ + a + b$, $\frac{\partial T}{\partial X}(0, 0) = a + b \neq 0$ d'où le point $(0 : 1 : 0)$ est un point infini non-singulier.

– $(1 : 1 : 0)$, $X = Y = 1$, nous avons l'équation suivante : $T(Z) = a[Z+Z+Z+Z^3] = aZ[1+Z^2]$.
$\frac{\partial T}{\partial Z} = a[1 + Z^2]$, $\frac{\partial T}{\partial Z}(0, 0) = a \neq 0$ d'où le point $(1 : 1 : 0)$ est un point infini non-singulier.

6.2.4 Equivalence birationnelle

Théorème 6.2.2. . *Soit $\mathbb{K}$ un corps de charactéristique 2 et $a,\ b \in \mathbb{K}$. Toute courbe d'équation affine $a[x^2+y^2+xy+1]+(a+b)[x^2y+y^2x] = 0$ avec $ab(a+b) \neq 0$ est birationnellement équivalente à une courbe elliptique de la forme $v^2 + v\left[\frac{1+au}{a+b}\right] = u\left[\frac{a}{a^2+b^2} + \frac{ab^2}{a^2+b^2}u^2\right]$*

via la fonction $\varphi : (x, y) \longmapsto (u, v)$, avec

$$\begin{cases} u = \dfrac{1}{a + (a + b)x} \\ \\ v = \dfrac{y}{a + (a + b)x} \end{cases} \iff \begin{cases} x = \dfrac{1 + au}{(a + b)u} \\ \\ y = \dfrac{v}{u} \end{cases}$$

Preuve :

a) Supposons que $v^2 + v\left[\frac{1+au}{a+b}\right] = u[\frac{a}{a^2+b^2} + \frac{ab^2}{a^2+b^2}u^2]$ et prouvons que $a[x^2 + y^2 + xy + 1] + (a + b)[x^2y + y^2x] = 0$.

Soit $H(x, y) = a[x^2 + y^2 + xy + 1] + (a + b)[x^2y + y^2x]$. Nous avons

$$\begin{aligned} H(x,y) &= a\left[\frac{1+a^2u^2}{(a^2+b^2)u^2} + \frac{v^2}{u^2} + \frac{v(1+au)}{(a+b)u^2} + 1\right] + (a+b)\left[\frac{v(1+a^2u^2)}{(a^2+b^2)u^3} + \frac{v^2(1+au)}{(a+b)u^3}\right] \\ &= a[(1+a^2u^2)u + uv^2(a^2+b^2) + uv(a+b)(1+au) + u^3(a^2+b^2)] + (a+b)[v(1+a^2u^2) + \\ &\quad v^2(a+b)(1+au)] \\ &= \frac{u(a+a^3u^2)}{a^2+b^2} + auv^2 + auv\frac{1+au}{a+b} + au^3 + v\frac{1+a^2u^2}{a+b} + v^2(1+au) \\ &= v^2 + v[\frac{1+au}{a+b}] + u[\frac{a}{a^2+b^2} + \frac{ab^2}{a^2+b^2}] \\ &= 0 \end{aligned}$$

b) Supposons que $a[x^2 + y^2 + xy + 1] + (a + b)[x^2y + y^2x] = 0$ et prouvons que $v^2 + v[\frac{1+au}{a+b}] = u[\frac{a}{a^2+b^2} + \frac{ab^2}{a^2+b^2}u^2]$.

Soit $G(u,v) = v^2 + v[\frac{1+au}{a+b}] + u[\frac{a}{a^2+b^2} + \frac{ab^2}{a^2+b^2}u^2]$. Nous avons :

$$G(u,v) = \frac{y^2}{[a+(a+b)x]^2} + \frac{y}{a+(a+b)x}[\frac{1+\frac{a}{a+(a+b)x}}{a+b}] + \frac{1}{a+(a+b)x}$$
$$[\frac{a}{a^2+b^2} + \frac{ab^2}{a^2+b^2} \times \frac{1}{(a+(a+b)x)^2}]$$
$$= y^2(a+(a+b)x) + y(a+(a+b)x) \times [\frac{a+(a+b)x+a}{a+b}] + [\frac{a(a^2+(a^2+b^2)x^2)}{a^2+b^2} + \frac{ab^2}{a^2+b^2}]$$
$$= ay^2 + (a+b)xy^2 + axy + (a+b)x^2y + a + ax^2$$
$$= a[x^2+y^2+xy+1] + (a+b)[x^2y+xy^2]$$
$$= 0$$

□

Corollaire 6.2.1. ***(Version projective)***
Soit $\mathbb{K}$ un corps de caractéristique 2 et $a,\ b \in \mathbb{K}$. Toute courbe d'équation projective $a[X^2Z + Y^2Z + XYZ + Z^3] + (a+b)[X^2Y + Y^2X] = 0$ avec $ab(a+b) \neq 0$ est birationnellement équivalente à une courbe de la forme :

$$V^2W + VW\left[\frac{W+aU}{a+b}\right] = U\left[\frac{aW^2}{a^2+b^2} + \frac{ab^2}{a^2+b^2}U^2\right]$$

via le changement de variable suivant :

$$\begin{cases} U = \dfrac{Z}{a+b} \\ V = \dfrac{Y}{a+b} \\ W = X + \dfrac{aZ}{a+b} \end{cases} \iff \begin{cases} X = aU + W \\ Y = (a+b)V \\ Z = (a+b)U \end{cases}$$

Preuve : Se référencer à la preuve précédente.

□

6.3 Universalité du modèle et loi d'addition

6.3.1 Universalité

Quand on introduit une nouvelle forme de courbe elliptique, il est important d'étudier combien de "bonnes" courbes sont isomorphes au nouveau modèle.

Théorème 6.3.1. *Soit $\mathbb{K} = \mathbb{F}_{2^l}$ avec $l \geq 5$, toute courbe elliptique donnée par $y^2 + xy = x^3 + \alpha x^2 + \beta,\ \beta \neq 0$ est birationnellement équivalente à une courbe de la forme $a[x^2+y^2+xy+1] + (a+b)[x^2y+y^2x] = 0$.*

Preuve :

- $a[x^2+y^2+xy+1]+(a+b)[x^2y+y^2x]=0$ est birationnellement équivalente sur $\mathbb{F}_{2^l}$ à une courbe elliptique de la forme

$$v^2+v[\frac{1+au}{a+b}]=u[\frac{a}{a^2+b^2}+\frac{ab^2}{a^2+b^2}u^2]$$

via la fonction $\varphi_1:(x,y)\longmapsto(u,v)$, avec

$$\begin{cases} u=\dfrac{1}{a+(a+b)x} \\ \\ v=\dfrac{y}{a+(a+b)x} \end{cases} \iff \begin{cases} x=\dfrac{1+au}{(a+b)u} \\ \\ y=\dfrac{v}{u} \end{cases}$$

- Nous avons également $v^2+v[\frac{1+au}{a+b}]=u[\frac{a}{a^2+b^2}+\frac{ab^2}{a^2+b^2}u^2]$ est birationnellement équivalente à $v'^2+a_1u'v'=u'^3+a_2u'^2+a_4u'+a_6$ avec $a_1=\dfrac{a}{c(a+b)}$, $a_2=\dfrac{1}{a}$, $a_4=\dfrac{1}{a^2}+\dfrac{1}{b^2}$ et $a_6=\dfrac{1}{c^2(a^2+b^2)}+\dfrac{1}{a^3}$ avec $c^2=\dfrac{ab^2}{a^2+b^2}$, via la fonction $\varphi_2:(u,v)\longmapsto(u',v')$, avec

$$\begin{cases} u'=\dfrac{1}{a}+u \\ \\ v'=\dfrac{v}{c} \end{cases} \iff \begin{cases} u=\dfrac{1}{a}+u' \\ \\ v=cv' \end{cases}$$

On définit un autre changement de variables $\begin{cases} u'=a_1^2T \\ v'=a_1^3(Z+sT+\lambda) \end{cases}$ alors nous avons

$$\begin{aligned} a_1^6(Z^2+s^2T^2+\lambda^2)+a_1^6t(Z+sT+\lambda) &= a_1^6T^3+a_1^4a_2T^2+a_1^2a_4T+a_6Z^2+Tz \\ &= T^3+T^2[s^2+s+\frac{a_2}{a_1^2}]+T[\lambda+\frac{a_4}{a_1^4}]+\lambda^2+\frac{a_6}{a_1^6} \end{aligned}$$

Par identification : $\begin{cases} s^2+s+\dfrac{a_2}{a_1^2}=a_2' \\ \lambda+\dfrac{a_4}{a_1^4}=0 \\ \lambda^2+\dfrac{a_6}{a_1^6}=a_6'\Rightarrow a_6'=\dfrac{a_4^2}{a_1^8}+\dfrac{a_6}{a_1^6}=\dfrac{a_4^2+a_1^2a_6}{a_1^8} \end{cases}$

On définit $h^{-2}=\dfrac{a_2}{a_1^2}\Longrightarrow h=\dfrac{a_1}{\sqrt{a_2}}$. D'où nous avons $s^2+s+a_2'+h^{-2}=0$, $h^{-2}=\dfrac{a_2}{a_1^2}=\dfrac{c^2(a^2+b^2)}{a^3}$

$$\begin{aligned} a_6'=\left(\frac{a_4+a_1\sqrt{a_6}}{a_1^4}\right) &= \left(1+\frac{c^2(a^2+b^2)}{a^3}+\sqrt{1+\frac{c^2(a^2+b^2)}{a^3}}\right)\frac{c^2(a^2+b^2)}{a^3} \\ &= \left(1+h^{-2}+\sqrt{1+h^{-2}}\right)h^{-2} \end{aligned}$$

$\Longrightarrow h^2\sqrt{a_6'}=h^{-2}+h^{-1}\Longleftrightarrow h^{-2}+h^{-1}+h^2\sqrt{a_6'}=0.$

Prenons $t=h^{-1}$ alors $t^2+t+h^2\sqrt{a_6'}=0$

D'où $\begin{cases} s^2+s+a_2'+h^{-2}=0 \\ t^2+t+h^2\sqrt{a_6'}=0 \end{cases} \iff \begin{cases} \mathbf{Trace}(a_2'+h^{-2})=0 \\ \mathbf{Trace}(h^2\sqrt{a_6'})=0 \end{cases} \iff \begin{cases} \mathbf{Trace}(h^{-1})=Tr(a_2') \\ \mathbf{Trace}(h\sqrt[4]{a_6'})=0 \end{cases}$

Pour tous $\lambda,\ \pi\in\mathbb{F}_2$, on définit l'ensemble suivant :

$$L_{\lambda,\pi}=\{h\in\mathbb{F}_{2^l}^*:\mathbf{Trace}(h^{-1})=\lambda,\mathbf{Trace}(h\sqrt[4]{a_6'})=\pi\}$$

On note $|L|$ le cardinal de l'ensemble L et $|E|$ le cardinal de E.

Puisque $t^4\sqrt{a'_6}+t+1=0$ a au plus 4 racines, nous devons prouver que $L_{\mathbf{Trace}(a'_6),0}$ a au moins 5 éléments *i.e* $|L|_{\mathbf{Trace}(a'_6),0} \geq 5$ si $l \geq 5$.

C'est à dire prouvons que $|L|_{0,0} \geq 5$ et $|L|_{1,0} \geq 5$ si $l \geq 5$.

Nous avons $|L|_{0,0}+|L|_{1,0} = 2^{l-1}-1$. Par conséquent, comme h peut prendre toute valeur dans $\mathbb{F}^*_{2^l}$, alors $h\sqrt[4]{a'_6}$ prend également toute valeur dans $\mathbb{F}^*_{2^l}$. Nous déduisons que $|L|_{0,0}+|L|_{1,0}$ compte les éléments $h \in \mathbb{F}^*_{2^l}$ avec $\mathbf{Trace}(h)=0$. Maintenant, nous avons $|L|_{1,0}+|L|_{1,1} = 2^{l-1}$. Par conséquent, comme précédemment $|L|_{1,0}+|L|_{1,1}$ compte les éléments $h \in \mathbb{F}^*_{2^l}$ avec $\mathbf{Trace}(h)=1$.

$|L|_{0,0}+|L|_{1,0} = \dfrac{2^l}{2}-1 = 2^{l-1}-1$, $|L|_{1,0}+|L|_{1,1} = \dfrac{2^l}{2} = 2^{l-1}$. Maintenant calculons $|L|_{0,0}+|L|_{1,1}$.

Nous avons

$h \in L_{0,0}\cup L_{1,1} \iff \begin{cases} \mathbf{Trace}(h^{-1}) = 0 = \mathbf{Trace}(h\sqrt[4]{a'_6}) \\ \mathbf{Trace}(h^{-1}) = 1 = \mathbf{Trace}(h\sqrt[4]{a'_6}) \end{cases} \iff \mathbf{Trace}(h^{-1}) = \mathbf{Trace}(h\sqrt[4]{a'_6}) \iff$

$\mathbf{Trace}(h^{-1}+h\sqrt[4]{a'_6}) = 0$ si et seulement si nous avons deux possibilités pour x, c'est à dire (x *et* $x+1$) telles que $x^2+x+h^{-1}+h\sqrt[4]{a'_6} = 0 \iff h^2x^2+h^2x+h+h^3\sqrt[4]{a'_6} = 0 \iff (hx)^2+h(hx) = h^3\sqrt[4]{a'_6}+h \iff v^2+uv = u^3\sqrt[4]{a'_6}+u$ avec $v=hx$ et $u=h$.

Comme le théorème de Hasse implique qu'il existe $\delta = |E(\mathbb{F}^*_{2^l})| - 2^l - 1 \in [-2\sqrt{2^l}, 2\sqrt{2^l}]$, de plus le point $(0,0)$ et le point à l'infini ne peuvent pas vérifier l'équation précédente et deux points sur la courbe produisent le même h, alors

$|L|_{0,0}+|L|_{1,1} = (|E(\mathbb{F}^*_{2^l})|-2)/2 = (\delta+2^l+1-2)/2$

$|L|_{0,0}+|L|_{1,1} = 2^{l-1}+\dfrac{\delta-1}{2}$

$4|L|_{1,0} = 2(|L|_{0,0}+|L|_{1,0})+2(|L|_{1,0}+|L|_{1,1})-2(|L|_{0,0}+|L|_{1,1}) = 2(2^{l-1}-1)+2(2^{l-1})-2(2^{l-1}-\dfrac{\delta-1}{2}) = 2^l-(\delta+1)$

$4|L|_{0,0} = 4(2^{l-1}-1)-4|L|_{1,0} = 4(2^{l-1}-1)-(2^l-(\delta-1)) = 22^l-4-2^l+\delta+1 = 2^l+\delta-3$,

Comme $\delta \in [-2\sqrt{2^l}, 2\sqrt{2^l}] \Longrightarrow \delta \geq -2\sqrt{2^l}$,

alors $4|L|_{0,0} = 2^l+\delta-3 \Longrightarrow 4|L|_{0,0} \geq 2^l+-2\sqrt{2^l}-3 \Longrightarrow |L|_{0,0} \geq \dfrac{2^l-2\sqrt{2^l}-3}{4}$ et

$4|L|_{1,0} \geq 2^l+-2\sqrt{2^l}-1 \Longrightarrow |L|_{1,0} \geq \dfrac{2^l-2\sqrt{2^l}-1}{4}$

$\dfrac{2^l-2\sqrt{2^l}-1}{4} \geq \dfrac{2^l-2\sqrt{2^l}-3}{4} \geq 5$?

$\dfrac{2^l-2\sqrt{2^l}-3}{4} = \dfrac{(\sqrt{2^l}-1)^2-4}{4} \geq \dfrac{(\sqrt{2^5}-1)^2-4}{4} = 11.25 \geq 5$

Comme remarque finale, dans le but de transformer la courbe $z^2+zt = t^3+a'_2t+a'_6$ en $a[x^2+y^2+xy+1]+(a+b)[x^2y+y^2x]=0$, nous devons trouver h avec $\mathbf{Trace}(h^{-1}) = \mathbf{Trace}(a'_2)$ et $\mathbf{Trace}(h\sqrt[4]{a'_6}) = 0$, $h^{-2} = \dfrac{c(a^2+b^2)}{a^3} = \dfrac{b^2}{a^2}$, $\dfrac{b}{a} = h^{-1} = t_0$ où $t_0^2+t_0+h^2\sqrt{a'_6} = 0$, $t_0^2 = \dfrac{b^2}{a^2}$, fixer b et calculer $a = \sqrt{\dfrac{b}{t_0}}$ et fixer a ou calculer $b = \sqrt{at_0}$.

□

Théorème 6.3.2. . *Soit* $\mathbb{K}$ *un corps de charactéristique* 2 *and* $a,\ b \in \mathbb{K}$. *Toute courbe d'équation affine* $a[x^2+y^2+xy+1]+(a+b)[x^2y+y^2x]=0$ *avec* $ab(a+b) \neq 0$ *est birationnellement équivalente*

à la courbe

$z^2+tz=t^3+a'_2t^2+a'_6$ *avec* $a'_2=\dfrac{b^2}{a^2}$ *et* $a'_6=\dfrac{a^4+b^4}{a^8}b^4+\dfrac{a^2+b^2}{a^6}b^4$ *via la fonction* $\psi:(x,y)\longmapsto(t,z)$ *avec*

$$\begin{cases} t=\dfrac{b^2}{a^2}\left[\dfrac{(a+b)x}{a+(a+b)x}\right] \\ \\ z=\dfrac{b^2}{a^2}[\dfrac{(a+b)y+\dfrac{a^2+b^2}{a^2}[a+(a+b)x]}{a+(a+b)x}] \end{cases} \Longleftrightarrow \begin{cases} x=\dfrac{\dfrac{a^3}{a+b}t}{b^2+a^2t} \\ \\ y=\dfrac{\dfrac{a^3}{a+b}z+\dfrac{a+b}{a}b^2}{b^2+a^2t} \end{cases}$$

Preuve :

a) Supposons que $z^2+tz=t^3+a'_2t^2+a'_6$ et prouvons que $a[x^2+y^2+xy+1]+(a+b)[x^2y+y^2x]=0$.

Soit $H(x,y)=a[x^2+y^2+xy+1]+(a+b)[x^2y+y^2x]$, nous avons :

$$H(x,y)=a\left[\frac{\dfrac{a^6}{a^2+b^2}t^2}{(b^2+a^2t)^2}+\frac{\dfrac{a^6}{a^2+b^2}z^2+\dfrac{a^2+b^2}{a^2}b^4}{(b^2+a^2t)^2}+\frac{\dfrac{a^3}{a+b}t\left[\dfrac{a^3}{a+b}z+\dfrac{a+b}{a}b^2\right]}{(b^2+a^2t)^2}+1\right]+(a+$$

$$b)\left[\frac{\dfrac{a^6}{a^2+b^2}t^2\left[\dfrac{a^3}{a+b}z+\dfrac{a+b}{a}b^2\right]}{(b^2+a^2t)^3}+\frac{\dfrac{a^3}{a+b}t\left[\dfrac{a^6}{a^2+b^2}z^2+\dfrac{a^2+b^2}{a^2}b^4\right]}{(b^2+a^2t)^3}\right]$$

$$=a[t^2(b^2+a^2t)+z^2(b^2+a^2t)+\frac{a^4+b^4}{a^8}b^4(b^2+a^2t)+zt(b^2+a^2t)+$$

$$\frac{a^2+b^2}{a^4}b^2t(b^2+a^2t)+\frac{a^2+b^2}{a^6}(b^2+a^2t)^3]+$$

$$(a+b)\left[t^2(\frac{a^3}{a+b}z+\frac{a+b}{a}b^2)+\frac{a^3}{a+b}z^2t+\frac{b^4(a+b)(a^2+b^2)}{a^5}t\right]$$

$$=z^2[ab^2]+zt[ab^2]+t^3[ab^2]+t^2\left[ab^2+\frac{a^2+b^2}{a}b^2\right]+\frac{a^4+b^4}{a^7}b^6+\frac{a^2+b^2}{a^5}b^6$$

$$=z^2+zt+t^3+\frac{b^2}{a^2}t^2+\frac{a^4+b^4}{a^8}b^4+\frac{a^2+b^2}{a^6}b^4$$

$$=z^2+zt+t^3+a'_2t^2+a'_6$$

$$=0$$

b) Supposons que $a[x^2+y^2+xy+1]+(a+b)[x^2y+y^2x]=0$ et prouvons que $z^2+tz=t^3+a'_2t^2+a'_6$.

Soit $G(t,z)=z^2+tz+t^3+a'_2t^2+a'_6$, nous avons :

$$G(t,z)=\frac{b^4}{a^4}[\frac{(a^2+b^2)y^2+\dfrac{a^4+b^4}{a^4}[a^2+(a^2+b^2)x^2]}{(a+(a+b)x)^2}]$$

$$+\frac{b^4}{a^4}[\frac{(a+b)x}{a+(a+b)x}][\frac{(a+b)y+\ dfraca^2+b^2a^2(a+(a+b)x)}{a+(a+b)x}]+\frac{b^6}{a^6}\times\frac{(a^2+b^2)(a+b)x^3}{(a+(a+b)x)^3}$$

$$+\frac{b^6}{a^6}\left[\frac{(a^2+b^2)x^2}{(a+(a+b)x)^2}\right]+\frac{b^4}{a^4}(a^2+b^2)\left[\frac{a^2+b^2}{a^4}+\frac{1}{a^4}\right]$$

$$=y^2[a+(a+b)x]+(a^2+b^2)\frac{a^2+(a^2+b^2)x^2}{a^4}[a+(a+b)x]+x[y+(a+b)\frac{a+(a+b)x}{a^2}][a+$$

$$(a+b)x]+\frac{b^2}{a^2}(a+b)x^3+\frac{b^2}{a^2}x^2[a+(a+b)x]+\frac{b^2}{a^4}[a^2+(a^2+b^2)x^2][a+(a+b)x]$$

$$=ay^2+(a+b)xy^2+\frac{1}{a}+\frac{a+b}{a^2}x+\frac{a^2+b^2}{a^3}x^2+\frac{(a^2+b^2)(a+b)}{a^4}x^3+axy+(a+b)x^2y+$$

$$x+\frac{a+b}{a}x^2+\frac{a+b}{a}x^2+\frac{a^2+b^2}{a^2}x^3+\frac{b^2}{a^2}(a+b)x^3+\frac{b^2}{a}x^2+\frac{b^2}{a^2}(a+b)x^3+\frac{b^2}{a^4}[a^3+a^2(a+b)x+a(a^2+b^2)x^2+(a^2+b^2)(a+b)x^3]$$
$$= a[x^2+y^2+xy+1]+(a+b)[x^2y+y^2x]$$
$$= 0$$

□

Corollaire 6.3.1. ***(Version projective)***. *Soit* $\mathbb{K}$ *un corps de charactéristique* 2 *et* $a,\ b \in \mathbb{K}$. *Toute courbe d'équation projective* $a[X^2Z+Y^2Z+XYZ+Z^3]+(a+b)[X^2Y+Y^2X]=0$ *avec* $ab(a+b)\neq 0$ *est birationnellement équivalente à la courbe d'équation* $V^2W+UVW=U^3+a_2'U^2W+a_6'W^3$ *avec* $a_2'=\frac{b^2}{a^2}$ *et* $a_6'=\frac{a^4+b^4}{a^8}b^4+\frac{a^2+b^2}{a^6}b^4$ *par*

$$\begin{cases} U=\dfrac{b^2(a+b)}{a^2}X \\ V=\dfrac{b^2}{a^2}[(a+b)Y+\dfrac{a^2+b^2}{a^2}(aZ+(a+b)X)] \\ W=aZ+(a+b)X \end{cases} \iff \begin{cases} X=\dfrac{a^3}{a+b}U \\ Y=\dfrac{a^3}{a+b}V+\dfrac{a+b}{a}b^2W \\ Z=a^2U+b^2W \end{cases}$$

Preuve : Se référencer à la preuve précédente.

□

6.3.2 Loi d'addition

- **Elément neutre :** Dans le corollaire 6.1.1, nous avons

$$\begin{cases} U=\dfrac{Z}{a+b} \\ V=\dfrac{Y}{a+b} \\ W=X+\dfrac{aZ}{a+b} \end{cases} \iff \begin{cases} X=aU+W \\ Y=(a+b)V \\ Z=(a+b)U \end{cases}$$

et le point à l'infini est $P_\infty=(0:1:0)$ de la courbe elliptique de la forme $V^2W+VW\left[\frac{W+aU}{a+b}\right]=U\left[\frac{aW^2}{a^2+b^2}+\frac{ab^2}{a^2+b^2}U^2\right]$.
L'élément neutre est le point $\varphi^{-1}(P_\infty)=\varphi^{-1}(0:1:0)=(0:a+b:0)=(0:1:0)$.

- **Elément symétrique :** si $P=(x,y)$ est un point sur la courbe $(\mathcal{E})$, nous avons $-P=\varphi^{-1}(-\varphi(P))$, et la courbe $v^2+v\left[\frac{1+au}{a+b}\right]=u\left[\frac{a}{a^2+b^2}+\frac{ab^2}{a^2+b^2}u^2\right]$, alors, nous avons $-\varphi(P)=-(u,v)=\left(u,v+\frac{1+au}{a+b}\right)$.
D'où l'élément symétrique est $-P=(x,x+y)$ sur $(\mathcal{E})$.

- **Loi d'addition :** soit $y = \alpha x + \beta$ l'équation de la droite (PQ) où $P = (x_P, y_P)$ et $Q = (x_Q, y_Q)$ sont deux points de la courbe $(\mathcal{E})$. On définit $P+Q = R$ où $R = (x_R, y_R)$ et $-R = (x_R, x_R+y_R)$ est le troisième point d'intersection entre la droite et la courbe.
  Nous avons $a[x^2 + (\alpha x + \beta)^2 + x(\alpha x + \beta) + 1] + (a + b)[x^2(\alpha x + \beta) + (\alpha x + \beta)^2 x] = 0$ alors $[(a + b)(\alpha + \alpha^2)]x^3 + [a(1 + \alpha + \alpha^2) + \beta(a + b)]x^2 + [a\beta + \beta^2(a + b)]x + a(\beta^2 + 1) = 0$.
  D'où $x_P + x_Q + x_R = \dfrac{a(1 + \alpha + \alpha^2) + \beta(a + b)}{(a + b)(\alpha + \alpha^2)}$
  Ainsi :
  $$\begin{cases} x_R = x_P + x_Q + \dfrac{a(1 + \alpha + \alpha^2) + \beta(a + b)}{(a + b)(\alpha + \alpha^2)} \\ y_R = \alpha x_R + \beta \end{cases}$$
  avec $\alpha = \dfrac{y_P + y_Q}{x_P + x_Q}$ et $\beta = y_P + \alpha x_P$

Conclusion

Nous avons proposé avec succès une nouvelle courbe elliptique binaire $(\mathcal{E})$. Pour les études ultérieures, nous devons étudier si la loi d'addition est unifiée et complète.

Ainsi notre courbe $(\mathcal{E})$ n'est pas pour le moment utilisable en cryptographie en l'état.

Conclusion et Perspectives de recherche

Conclusion

Dans cette thèse, nous avons proposé quelques solutions aux propblèmes liés à l'amélioration de l'efficacité et de la sécurité du cryptosystème El Gamal et de l'échange de clé Diffie-Hellman.

Dans la première partie, nous avons proposé un nouveau algorithme permettant d'améliorer le cryptosystème El Gamal notamment la réduction de la taille des clés de déchiffrement et des variantes de schémas de signature. Nous avons aussi "masqué" le générateur pour qu'il ne puisse pas être transmis en clair via le réseau si nécessaire.

Dans la deuxième partie, nous avons "attaqué" le protocole "SDH-DSA-KE" et proposé une nouvelle version qui l'améliore et qui peut s'implémenter sur les courbes elliptiques.

Enfin dans la troisième partie, on présente une nouvelle version de courbe elliptique binaire ($\mathcal{E}$) birationnellement équivalente à toute courbe de la forme $y^2 + xy = x^3 + ax + b$ avec a et $b \neq 0$ sur ($\mathbb{F}_{2^m}$).

Perspectives de recherche

- Finir le travail sur la nouvelle courbe elliptique qui ouvrira énormément de sujets de recherche (hachage, couplage, multiplication, extraction aléatoire, $\cdots$) ;
- Faire la preuve de sécurité de la signature ;
- Implémenter notre variante El Gamal sur les courbes elliptiques ;
- Implémenter notre protocole d'échange dans un protocole réseau.
- Etudier les courbes elliptiques sur l'anneau $\frac{K[t]}{(t^2 - t)}$, où K est un corps (noter que récemment les courbes elliptiques ont été étudiées sur l'anneau des nombres duaux $\frac{K[t]}{(t^2)}$).

Annexe

Code source en java du chiffrement sur $\mathbb{Z}/p\mathbb{Z}$

```
    import java.util.Random ;
import java.io.BufferedReader ;
import java.io.IOException ;
import java.io.InputStreamReader ;
import java.math.BigInteger ;
import java.security.SecureRandom ;

    public class Chiffrement {
private final static SecureRandom random = new SecureRandom() ;

    ********** Génère un nombre aléatoire trés grand **********
public static BigInteger bigRandom(int nbBits){
Random rnd=new Random() ;
BigInteger tmp= new BigInteger(nbBits,rnd) ;
return tmp ;
}
********** Trouve les deux diviseurs du nombre donné en entrée *******
public static BigInteger [] getqm(BigInteger p){
p = p.subtract(BigInteger.ONE) ;
BigInteger two = new BigInteger("2") ;
BigInteger neg = new BigInteger("-1") ;
BigInteger [] rt =BigInteger.ZERO, BigInteger.ZERO ;
if (p.mod(two).compareTo(BigInteger.ZERO) != 0) { rt[0] = neg ; rt[1] = neg ;
return rt ;
}
BigInteger divisor = p.divide(two) ;
BigInteger counter = BigInteger.ONE ;
while (divisor.mod(two).compareTo(BigInteger.ZERO)==0){
counter = counter.add(BigInteger.ONE) ;
```

```
divisor = divisor.divide(two) ;
}
rt[0] = counter ; rt[1] = divisor ;
return rt ;
}
```

——— Mise en place de l'algorithme millerRabin qui permet de deterniner la primalitéd'un entier de grande taille ————.

```
public static boolean millerRabin(BigInteger pval) {
BigInteger [] qandm = getqm(pval) ;
BigInteger qval =qandm[0] ;
BigInteger neg = new BigInteger("-1") ;
if (qval.compareTo(neg)==0)return false ;
BigInteger bval = bigRandom(pval.bitLength()) ;
BigInteger mval =qandm[1] ;
BigInteger two = new BigInteger("2") ;
BigInteger pminusone = pval.subtract(BigInteger.ONE) ;
if (bval.modPow(mval,pval).compareTo(BigInteger.ONE)==0) return true ;
BigInteger j = BigInteger.ZERO ;
BigInteger indexval = mval ;
while (j.compareTo(qval)<0){
if (pminusone.compareTo(bval.modPow(indexval,pval))==0)return true ;
indexval = indexval.multiply(two) ;
j = j.add(BigInteger.ONE) ;
}
return false ;
}
```

——————— FONCTION PRINCIPALE ———————-

```
public static void main(String[] args) throws IOException{
————————- INITIALISATION DES PARAMETRES ————————-
BigInteger a,b,c,i,j,p, h, d, g, s, r, t, k, kd, g0, p0, rs,v1,v2 ;
BigInteger alpha, gama, lamda, delta ;
BigInteger un= new BigInteger("1") ;
BigInteger deux = new BigInteger("2") ;
BigInteger x=null, y=null ;
————————- GENERATION DES CLES ————————
————————Génération du modulo p ————————
j = un ;
```

```
do{
d = BigInteger.probablePrime(128, random) ;
}while(millerRabin(d)==false) ;
do{
p = deux.multiply(j).multiply(d).add(un) ;
j = j.add(un) ;
}while(millerRabin(p)==false) ;
————————- Le générateur g ————————-
g = new BigInteger(32, random) ;
do{
h = g.modPow((p.subtract(BigInteger.ONE)).divide(d),p) ;
g = g.add(un) ;
}while(h.intValue() !=1) ;
————————- génération de k et calcul de k*d ————————
do{
k = bigRandom(32) ;
}while(k.longValue()<deux.pow(31).longValue()) ;
kd = k.multiply(d) ;
————————- génération de s et calcul des paramètres r et t ————————
do{
s = bigRandom(32) ;
t = kd.remainder(s) ;
}while(s.longValue()<deux.pow(31).longValue()||s.gcd(d).intValue() !=1
    ||t.intValue()<deux.pow(2).intValue()||t.intValue()>deux.pow(16).intValue()) ;

    ———————— La Cle Privée ————————
r = kd.divide(s) ;

    ———————— Calcul de GAMA et DELTA ————————
gama = g.modPow(s,p) ;
delta = g.modPow(t,p) ;

    ————————————————- CHIFFREMENT ————————————————-
————————- Permet de saisir le texte avec le clavier ————————
BufferedReader message1 =new BufferedReader(new InputStreamReader(System.in)) ;
System.out.println("Veuillez saisir le texte à chiffrer : ") ;
String plaintext = message1.readLine() ;
byte[] plaintextByte = plaintext.getBytes() ;
```

```
BigInteger plaintextBigInteger = new BigInteger(plaintextByte);
———————— le texte clair en biginteger ————————————
BigInteger m = plaintextBigInteger;
————————— génération de alpha pour le chiffrement tel que pgcd(alpha,p-1)=1 ——————
do{
alpha = bigRandom(32);
}while(alpha.longValue()<deux.pow(31).longValue()||alpha.gcd(d).intValue()!=1);

    ————————- calcul de m1 = γ^α mod p ————————————
BigInteger m1 = gama.modPow(alpha,p);
————————— calcul de λ = δ^α mod p ————————————
lamda = delta.modPow(alpha,p);

    ———————— calcul de m2 = λ * m ————————————————-
BigInteger m2 = lamda.multiply(m).mod(p);

    ———————————— déchiffrement : m1^r * m2 ————————————————
BigInteger m3 = m1.modPow(r,p);
BigInteger dechiffre = (m2.multiply(m3)).mod(p);

    ———————— Conversion en byte puis en chaine du déchiffé obtenu en BigInteger ————————
byte[]dechiffreByte = dechiffre.toByteArray();
String dechiffreString = new String(dechiffreByte);

    ———————————— AFFICHAGE DES RESULTATS ————————————————
//System.out.println("PARAMETRES");
System.out.println("P : " + p);
System.out.println("Ordre de g : d = " + d);
System.out.println("Générateur g : " + g);
System.out.println("Voici la valeur de k : " + k);

    System.out.println("CLE PUBLIQUE");
System.out.println("Gama : " + gama);
System.out.println("Delta : " + delta);

    System.out.println("CLE PRIVEE");
System.out.println("S : " + s);
System.out.println("R : " + r);
```

```
System.out.println("T : " + t);

    System.out.println("Voici le paramètre aléatoire alpha : " + alpha);
System.out.println("Le message m en BigInteger : " + m);
System.out.println("Le chiffré m1 : " + m1);
System.out.println("Le chiffré m2 : " + m2);
System.out.println("Utlisation de la clé privée : m3 = m1^r : " + m3);
System.out.println("Le déchiffré en BigInteger : m2 * m3 " + dechiffre);
System.out.println("Le déchiffré en String : " + dechiffreString);
}

    }
```

Bibliographie

Bibliographie

[1] Antoine Joux and Kim Guyen *Separating Decision Diffe-Hellman to Diffe Hellmann in cryptographie groups* http ://citeseer.ist.psu.edu (2006)

[2] A. Arazi, "*Inegrating a key cryptosystem into the digital signature standard*", Electron. Lett., vol. 29, pp. 966-967, Nov. 1993.

[3] R. Avanzi, H. Cohen, C. Doche, G. Frey, T. lange, K. Nguyen, and F. Vercauteren. *Handbook of elliptic and hyperelliptic curve cryptography.* Chapman and Hall, 2006.

[4] M. Bellare and P. Rogaway, *Random oracles are practical : a paradigm for designing efficient protocols.* Proceedings of the First Annual Conference on Computer and Communications Security, ACM, 1993.

[5] D.J. Bernstein, T. Lange, and R.R. Farashahi. *Binary edwards curves.* Cryptology ePrint Archive, Report 2008/171, 2008.

[6] D. Bleichenbacher *Generating Elgamal Signature without knowing the secret key* Presented at Eurocrypt 96 and revised April 16, 1996.

[7] D. Boneh. *The Decision Diffie-Hellman problem.* In Proceedings of the Third Algorithmic Number Theory Symposium, volume 1423 of LNCS, pages 4863. Springer- Verlag, (1998).

[8] R. Canetti, O. Goldreich, and S. Halevi. *The random oracle methodology, revisited.* J. ACM, $51(4) : 557 - 594$ (electronic), 2004.

[9] C. De Cannière and B. Preneel, "T*rivium - A Stream Cipher Construction In-espired by Block Cipher Design Principles*," to be uploaded to http ://www.ecrypt.eu.org/stream soon.

[10] J. Devigne and M. Joye *Binary Huff Curves* A. Kiayias, Ed., Topics in Cryptology. CT-RSA 2011, vol. 6558 of Lecture Notes in Computer Science, pp. 340-355, Springer, 2011.

[11] W. Diffie and M. E. Hellman. *New directions in cryptography. IEEE Trans. Inform. Theory*, IT-22 :644-654, Nov 1976.

[12] FIPS (Federal Information Processing Standard), *Advanced Encryption Standard (AES)* (FIPS PUB 197). 2. Category of Standard. Computer Security Standard, Cryptography. 3. Explanation. src.nist.gov/publications/fips/fips197/fips-197.pdf, 26 nov. 2001.

[13] FIPS, *Data Encryption. Standard (DES)*(FIPS 46-3). csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf ?,

[14] FIPS, *Digital Signature Standard (DSS)*. National Institute of Standards and Technology (NIST), 1994.

[15] FIPS, National Institute of Standards and Technology (NIST), *Secure Hash Algorithm (SHA)*. SP. csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf, Mar 4, 2012.

[16] H. M. Edwards. *A normal form for elliptic curves*. Bulletin of the American Mathematical Society 44(2007), 2007.

[17] T. E. Gamal. *A public key cryptosystem and a signature scheme based on discrete logarithms*. IEEE Trans. Inform. Theory, 31 :469-472, 1985.

[18] S. Goldwasser, S. Micali and R. Rivest, *A digital signature scheme secure against adaptive chosen-message attacks*, SIAM Journal of computing, 17(2), pp. 281-308, April 1988.

[19] J. Groth *Simulation-Sound NIZK Proofs for a Practical Language and Constant Size Group Signatures*. In proceedings of ASIACRYPT'06, LNCS 4284 (2006) 444-459.

[20] L. Harn, Mehta, and W. J. Hsin. "*Integratin Diffie-Hellman key exchange into a digital signature algorithm (DSA),*" IEEE Commun. Lett., vol. 8, pp. 198-200, Mar. 2004.

[21] P. Horster, H. Petersen and M. Michels *Meta-ElGamal signature schemes* (ACM CCS 1994).

[22] G.B. Huff : *Diophantine problems in geometry and elliptic ternary forms*. Duke Math. J. 15, 443-453 (1948)

[23] I. R. Jeong, J. O Kwon and D. H. Lee *Strong Diffie-Hellman DSA Key Exchange* IEEE Communications Letters, Vol 11, NO 5 May 2007, pp 432-433.

[24] M. Joye, M. Tibouchi, D. Vergnaud : *Huff's model for elliptic curves*. In : Hanrot, G., Morain, F., Thome, E. (eds.) Algorithmic Number Theory (ANTS-IX). Lecture Notes in Computer Science, vol. 6197, pp. 234-250. Springer (2010)

[25] N. Koblitz. *Elliptic curve cryptosystems*. Math. Comp., 1987.

[26] N. Koblitz. *Hyperelliptic cryptosystems*. Journal of Cryptography, 1989.

[27] P. Kocher. *Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems*. In Advances in cryptology - CRYPTO, volume 1109 of LNCS, pages 104-113. Springer, August 1996.

[28] H. Krawczyk, *HMQV : a high-performance secure Diffie-Hellman Protocol*, in Proc. CRYPTO'05, pp. 546-566.

[29] L. Law, A. Menezes, M. Qu, J. Solinas, and S. Vanstone, *An efficient Protocol for Authenticated Key Agreement*, Designs, Codes and Cryptography, 28, 119-134, 2003.

[30] A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[31] R. J. McEliece (January and February 1978). "*A Public-Key Cryptosystem Based On Algebraic Coding Theory*". DSN Progress Report. 42-44 : 114. Bibcode :1978DSNPR..44..114M. Retrieved 21 February 2009

[32] V. Miller. *Use of elliptic curves in cryptography.* In Proc. of Crypto '85, volume 218 of Lecture Notes in Computer Science, pages 417-426. IACR, Springer-Verlag, 1987.

[33] R. Needham and M. Schroeder. Using Encryption for Authentication in Large Networks of Computers. In CSL, number 78-4, 1978.

[34] K. Nyberg and R. A. Rueppel, "*Weaknesses in some recent key agreement protocols,*" Electron. Lett., vol. 30, pp. 26-27, Jan. 1994.

[35] Jr., W.D., Peeples : *Elliptic curves and rational distance sets.* Proc. Am. Math. Soc. 5, 29-33 (1954)

[36] R. C. -W. Phan, "*Fixing the integrated Diffie-Hellman-DSA key exchange protocol,*" IEEE Commun. Lett., vol. 9, pp. 570-572, June 2005.

[37] D. Pointcheval and J. Stern *Security Proofs for Signature Schemes* Advances in Cryptology Proceedings of EUROCRYPT '96 (may 12-16, 1996, Zaragoza, Spain) U. Maurer, Ed. Springer-Verlag, LNCS 1070, pages 387-398.

[38] D. Pointcheval *Provable Security for Public Key Schemes*
Advanced Course on Contemporary Cryptology, pages 133-189, June 2005. http ://www.di.ens.fr/ pointche/pub.php ?reference=Po04

[39] G. Poupard and J. Stern *Short proof of knowlegde for factoring* Volume 1751 Lecture Notes in Computer Science(Springer Berlein/Heidelberg) (2004) pages 147-166

[40] D. K. Rappe *Hommomorphic crytptosystems and there applications* Phd Thesis Dortemund University (2004)

[41] R. L. Rivest, A. Shamir, and L. M. Adleman. *A method for obtaining digital signatures and public-key cryptosystems.* Communications of the ACM, 21(2) :120-126, 1978.

[42] J. Silvermann. *The Arithmetique of Elliptic Curves.* Springer, 1986.

[43] Solinas, J. : *An improved algorithm for arithmetic on a family of elliptic curves*, Advances in Cryptology Crypto, 97, 357-371(1997)

[44] TAI Suiyan , LUO Ping , PENG Xiaoning , WANG Daoshun *Weak-Keys in Public Key Cryptosystems Based on Discrete Logarithms* (2005) http ://scholar.ilib.cn/A-qhdxxb-e200505010.html

[45] D. Vergnaud *Course "Provable Security in Public-Key Cryptography"* Lecture 1, 2, 3 and 4 in June 2010 at Dakar available at http ://www.di.ens.fr/ damien (Ecole Normale Superieure, C.N.R.S. I.N.R.I.A. (France)).