UNIVERSITÉ DE LIMOGES
ÉCOLE DOCTORALE Science – Technologie – Santé
FACULTÉ des Sciences et Techniques
Département Mathématiques et Informatique – XLIM

Thèse N° 60-2006

Thèse

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE LIMOGES

Discipline : mathématiques et applications

présentée et soutenue

par

Guilhem CASTAGNOS

le 3 octobre 2006, à 14 h 30, en salle des séminaires
du bâtiment de mathématiques

Quelques schémas de cryptographie asymétrique probabiliste

Thèse dirigée par François ARNAULT et Thierry BERGER

Jury

Président

Moulay BARKATOU, Professeur à l'Université de Limoges

Rapporteurs

Jean-Marc COUVEIGNES, Professeur à l'Université Toulouse II
François MORAIN, Professeur associé à l'École Polytechnique, HDR

Examinateurs

François ARNAULT, Maître de conférences à l'Université de Limoges
Thierry BERGER, Professeur à l'Université de Limoges
David POINTCHEVAL, Chargé de Recherche à l'École Normale Supérieure, HDR

Remerciements

Mes remerciements vont en premier lieu à François Arnault pour m'avoir confié ce sujet de recherche et m'avoir soutenu par ses conseils et ses idées durant ces trois années, toujours dans la bonne humeur. Je remercie également mon directeur de thèse « officiel », Thierry Berger, parfaitement complémentaire de François, pour ses nombreux conseils et pour avoir su surmonter certaines « formalités » administratives de début et de fin de thèse.

C'est avec grand plaisir que j'exprime ma gratitude envers Jean-Marc Couveignes qui après m'avoir, il y a quelques années, fait découvrir la cryptographie, a accepté de rapporter cette thèse. Je remercie également François Morain d'avoir effectué ce travail de rapporteur malgré les courts délais impartis.

Mes remerciements vont aussi à David Pointcheval pour avoir témoigné de l'intérêt pour mon travail en acceptant de siéger dans mon jury. Je remercie également Moulay Barkatou, que j'ai côtoyé dans le bâtiment de mathématiques de Limoges pendant ces trois années, d'avoir accepté de prendre part à ce jury.

Pendant la durée de cette thèse, j'ai partagé, avec plusieurs personnes, l'atmosphère détendue du fameux « bureau vert ». Dans l'ordre chronologique, vient d'abord Matthieu Le Floc'h, maintenant parti sous des cieux qu'il trouve plus cléments, qui malgré la barrière de la culture régionale a facilité mon implantation à Limoges. Ensuite, vient Thomas Cluzeau, qui après de multiples allers-retours a fini par fixer ses posters à Limoges, dans un autre bureau. Puis vient Mikaël Lescop, avec qui j'ai partagé bien des discussions extra-mathématiques, au cours de longs repas. Je lui souhaite une bonne adaptation à sa nouvelle carrière. Bonne continuation aussi à Mohsen Asghari, parti également pour une nouvelle fonction, ainsi qu'à Pierre-Louis Cayrel et Ahmed Aït-Mokhtar qui partagent ce bureau avec moi, depuis maintenant une année.

Je tiens également à manifester aux autres personnes du laboratoire qui m'ont soutenu, aidé, encouragé ou tout simplement témoigné de la sympathie, mes remerciements, et plus particulièrement à Anne Bellido, Laurent Dubreuil, Philippe Gaborit, Sylvie Laval, Nicolas Le Roux, Samuel Maffre, Henri Massias, Abdelkader Necer, Ayoub Otmani, Olivier Ruatta, Marc Rybowicz, Pascale Sénéchaud, Patricia Vareille, Yolande Vieceli (toujours enthousiaste et disponible), Stéphane Vinatier (arrivé à Limoges en même temps que moi et qui a toujours un créneau libre pour partager un moment de détente) et à Jacques-Arthur Weil.

Je salue aussi les « PICSIs », présents dans le laboratoire pendant une bonne partie de

ma thèse, et plus précisément le toulousain Olivier Rossi, camarade d'études et de soirées et Constantin Yamkoudougou.

Pour finir, je remercie mes parents et ma sœur qui ont fait le déplacement dans le grand Nord pour assister à ma soutenance et Aude pour son soutien de tous les instants.

TABLE DES MATIÈRES

INTRODUCTION

Les systèmes asymétriques probabilistes ont été introduits en 1984 par Goldwasser et Micali dans [GM84]. Dans ces systèmes, un message a plusieurs chiffrements possibles, en fonction d'un aléa. En plus de la notion de sécurité traditionnelle, apparaît la notion de sécurité sémantique. Intuitivement, il s'agit étant donné deux messages et le chiffré c de l'un de ces messages, qu'un adversaire ne puisse déterminer duquel des deux messages c est le chiffré.

Depuis le système de Goldwasser et Micali, d'autres systèmes probabilistes partageant le même principe ont été proposés, tout d'abord par Benaloh dans [Ben88], puis Naccache et Stern dans [NS98], Okamoto et Uchiyama dans [OU98a] et enfin le système le plus abouti, celui de Paillier dans [Pai99]. Tous ses systèmes utilisent les quotients de $\mathbf{Z}$ afin de construire des schémas dont la sécurité sémantique est basée sur la reconnaissance de certaines puissances. De plus ces systèmes ont une structure algébrique très riche et sont en particulier homomorphiques, propriété très recherchée pour les applications au vote électronique.

À partir des idées de Paillier, de multiples variantes très intéressantes sont apparues. Certaines, non homomorphiques (cf. [CGHGN01]), accélèrent le chiffrement et rendent le système proche d'un RSA probabiliste. D'autres variantes utilisent les courbes elliptiques. Outre le défi de ce changement de cadre qui nécessite d'étudier des courbes elliptiques définies sur des anneaux finis et de trouver de nouvelles formules d'additions (cf. [Gal02]), cela permet aussi de construire un système assez compétitif, avec un chiffrement comparable au système El Gamal elliptique (cf. [GMMV02]).

Les courbes elliptiques ont souvent été utilisées comme alternative aux groupes quotients de $\mathbf{Z}$. Que ce soit, comme on vient de le voir, pour la cryptographie, ou comme outils pour la primalité et la factorisation.

Un autre champ d'investigation pour construire des systèmes de chiffrement est celui des quotients de corps quadratiques. Tout comme les courbes elliptiques, de tels quotients ont joué un rôle pour la primalité (test de Lucas) et la factorisation (méthode $p+1$). Par contre, ils n'ont été que rarement utilisés pour la cryptographie. On peut tout de même citer le cryptosystème LUC proposé par Smith et Lennon dans [SL93]. La raison de ce manque d'intérêt est peut être l'utilisation abusive des suites de Lucas qui masque le fait que l'on travaille dans des groupes finis très proches des quotients de $\mathbf{Z}$. Les suites de Lucas permettent en fait de calculer de manière très efficace l'exponentiation dans ces groupes.

Les travaux de cette thèse visent d'une part à construire des fonctions trappe probabilistes (certaines généralisent les schémas cités ci-dessus) et d'autre part à appliquer ces

fonctions trappe dans les quotients de corps quadratiques afin de proposer des systèmes efficaces.

Dans un premier chapitre, on rappellera les propriétés des schémas de chiffrement asymétrique probabiliste. On s'intéressera formellement aux notions de sécurité qui sont associées à ces schémas. On énoncera en particulier une nouvelle formulation de la sécurité sémantique utilisée implicitement dans la littérature et on montrera son équivalence avec la formulation habituelle d'indistinguabilité.

Dans le deuxième chapitre, on introduira trois familles de fonctions trappe probabilistes. On utilisera des constructions génériques, certaines généralisant les cryptosystèmes cités précédemment.

La première famille de fonctions trappe présentée sera une famille de fonctions homomorphiques. Étant donnés un groupe abélien fini multiplicatif G et un entier k divisant $|G|$ tel que $|G|$ et $|G|/k$ soient premiers entre eux, on construira une fonction trappe de $\mathbf{Z}/k\mathbf{Z}$ dans G permettant d'établir un système homomorphique dont la sécurité sémantique est basée sur la reconnaissance des puissances k–ièmes dans G.

Le deuxième famille sera une famille de fonctions trappe probabilistes non homomorphiques. On généralisera la construction précédente en remplaçant le morphisme de G dans G qui à x associe x^k, qui permettait de construire des puissances k–ièmes, par une fonction trappe déterministe plus rapide à évaluer. On obtiendra ainsi une construction efficace pour produire un système probabiliste à partir d'un système non probabiliste. De plus, la sécurité de ce système sera fortement reliée à la sécurité de la fonction déterministe utilisée.

La troisième famille sera également une famille de fonctions trappe probabilistes non homomorphiques. La construction sera totalement différente des deux premières. À partir de deux fonctions trappe déterministes, permutations d'un même groupe, on montrera comme en construire une troisième qui, elle, sera probabiliste. La sécurité du système sera équivalente à celle des deux fonctions utilisées.

Dans le troisième chapitre on s'intéressera aux trois familles de groupes finis cités plus haut : les quotients de $\mathbf{Z}$, les groupes de points de courbes elliptiques définies sur des quotients de $\mathbf{Z}$ et enfin les groupes finis issus des quotients de corps quadratiques.

Soit n un entier strictement supérieur à 1. Le groupe $(\mathbf{Z}/n\mathbf{Z})^\times$ étant bien connu on se limitera, dans l'optique du chapitre suivant, à l'étude de la complexité des opérations dans $\left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times$ où s est un entier naturel.

On étudiera ensuite le groupe formé par l'ensemble des points d'une courbe elliptique définie sur $\mathbf{Z}/n\mathbf{Z}$. Ce type de groupe même s'il a été souvent utilisé n'a jamais été complètement détaillé. On verra qu'à l'aide du théorème chinois, on peut ramener l'étude aux courbes elliptiques définies sur $\mathbf{Z}/p^{s+1}\mathbf{Z}$ où p est un nombre premier et s est un entier naturel. Grâce au groupe formel d'une telle courbe, on arrivera à bien cerner sa structure de groupe et à fournir un système complet de formules d'additions.

On étudiera enfin le groupe des éléments de norme 1 d'un corps quadratique modulo n, noté $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. On observera que ce groupe a des propriétés très proches de celles des quotients de $\mathbf{Z}$. Ce sera aussi l'occasion de faire le lien entre les suites de Lucas et l'exponentiation des éléments de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. On s'apercevra ainsi que l'on obtient un outil

beaucoup plus performant que les courbes elliptiques en terme de complexité de calcul. On fera le lien entre le groupe $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ et le tore algébrique proposé en cryptographie dans [RS03].

Dans le dernier chapitre, on utilisera les fonctions trappe définies au deuxième chapitre avec les groupes finis étudiés dans le troisième.

On verra que les systèmes existants cités plus haut peuvent être considérés comme des instances de la première fonction trappe dans les quotients de $\mathbf{Z}$ et les courbes elliptiques. On proposera un nouveau système probabiliste homomorphique en utilisant cette fonction trappe dans les quotients de corps quadratiques. Ceci permettra d'obtenir un système beaucoup plus performant que ceux utilisant les courbes elliptiques.

Afin d'utiliser les deux autres fonctions trappe du deuxième chapitre, on introduira plusieurs fonctions trappe déterministes généralisant la fonction trappe RSA : en particulier la fonction KMOV en utilisant les courbes elliptiques et la fonction LUC en utilisant le groupe $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ introduit au troisième chapitre. On observera ainsi que l'introduction de ce groupe permet une description très naturelle du système LUC. On verra alors, que la deuxième fonction trappe du deuxième chapitre utilisée conjointement avec les fonctions RSA et KMOV permet de retrouver des systèmes existants. En l'utilisant avec la fonction LUC, on créera un nouveau système probabiliste performant. Ce système a été proposé dans [Cas]. On montrera aussi comment utiliser la troisième fonction trappe du deuxième chapitre à l'aide des fonctions LUC et RSA. On obtiendra ainsi un système très performant en chiffrement.

NOTATIONS

$a \leftarrow \mathscr{A}(b)$	L'algorithme $\mathscr{A}$ prend en entrée b et retourne a.
$c \leftarrow \mathrm{I}$	On tire au sort l'élément c de l'ensemble I avec distribution uniforme.
$\overline{b}$	Si b est un bit, $\overline{b} := b \oplus 1$.
$\overline{g}$	Suivant le contexte, $\overline{g}$ désigne la classe de g modulo une certaine relation d'équivalence.
$\lvert g \rvert$	Si g est un élément d'un groupe, $\lvert g \rvert$ désigne l'ordre de g.
$\lvert \mathrm{I} \rvert$	Si I est un ensemble (resp. un groupe), $\lvert \mathrm{I} \rvert$ désigne son cardinal (resp. son ordre).
$\lvert k \rvert_2$	Si k est un entier, $\lvert k \rvert_2$ désigne sa taille, *i.e.*, le nombre de bits intervenant dans son écriture binaire : $\lvert k \rvert_2 := \lfloor \log_2 k \rfloor + 1$.
$\langle g \rangle$	Si g est un élément d'un groupe, $\langle g \rangle$ désigne le sous-groupe qu'il engendre.
$\mathscr{R}^\times$	Si $\mathscr{R}$ est un anneau, $\mathscr{R}^\times$ désigne le groupe de ses éléments inversibles.
$\mathrm{A} \overset{\mathscr{P}}{\Longleftarrow} \mathrm{B}$	Il existe une réduction polynomiale du problème A au problème B.
$\mathrm{A} \overset{\mathscr{P}}{\Longleftrightarrow} \mathrm{B}$	Les problèmes A et B sont polynomialement équivalents.
$\mathrm{I} \xrightarrow{\sim} \mathrm{J}$	Il existe une bijection (resp. un isomorphisme) de l'ensemble I (resp. du groupe I) sur l'ensemble J (resp. vers le groupe J)
$\mathrm{I} \twoheadrightarrow \mathrm{J}$	Il existe une surjection (resp. un morphisme surjectif) de l'ensemble I (resp. du groupe I) sur l'ensemble J (resp. vers le groupe J)
$\mathrm{I} \hookrightarrow \mathrm{J}$	Il existe une injection (resp. un morphisme injectif) de l'ensemble I (resp. du groupe I) vers l'ensemble J (resp. vers le groupe J). Peut également désigner une inclusion.
entier RSA	Un entier n est un entier RSA s'il est le produit de deux grands nombres premiers distincts, tels que la factorisation de n ne puisse se faire en temps raisonnable.

CHAPITRE I

RAPPELS

Dans la première section, on définit la notion de schéma de chiffrement asymétrique probabiliste et ses propriétés. Dans la seconde section, on s'intéresse aux diverses notions de sécurité pour ses schémas. La plupart des définitions de ce chapitre sont inspirées de [BDPR98].

1. Schémas de chiffrement asymétrique probabiliste

Définitions

Dans le contexte d'un schéma de chiffrement asymétrique (ou à clef publique), un utilisateur diffuse une fonction de chiffrement tout en gardant secrète la fonction de déchiffrement correspondante. De cette manière, les personnes qui veulent lui envoyer des messages de manière chiffrée utilisent cette fonction publique. Cet utilisateur retrouve alors le message initial au moyen de la fonction secrète.

Si un message clair est toujours chiffré de la même manière, un attaquant va vite en tirer partie : il pourra par exemple voir si un même message est chiffré plusieurs fois et en déduire des informations. On voudra donc que le schéma de chiffrement soit probabiliste, c'est à dire que la fonction publique utilise un aléa afin qu'un message ait plusieurs chiffrés possibles. Ce n'est pas, par exemple, le cas du système RSA.

Plus formellement on a la définition suivante.

Définition I – 1. *Un schéma de chiffrement asymétrique probabiliste* Π *est la donnée d'un triplet de trois algorithmes* $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ *satisfaisant les trois propriétés suivantes.*

1. *L'algorithme* $\mathcal{K}$ *est appelé algorithme de génération de clef. C'est un algorithme probabiliste qui prend en entrée un entier naturel* k*, appelé paramètre de sécurité, donné en notation unaire, et qui retourne un couple* $(\mathrm{pk}, \mathrm{sk})$ *constitué d'une clef publique,* pk*, et d'une clef secrète,* sk*. On notera* $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathcal{K}(1^k)$*;*
2. *L'algorithme* $\mathcal{E}$ *est appelé algorithme de chiffrement. C'est un algorithme probabiliste qui prend en entrée une clef publique* pk *et un message clair,* m*, élément de* $\mathcal{M}$ *(l'espace des messages clairs). L'algorithme* $\mathcal{E}$ *retourne un chiffré,* c*, élément de* $\mathcal{C}$ *(l'espace des chiffrés). On notera* $c \leftarrow \mathcal{E}_{\mathrm{pk}}(m)$*;*

3. *L'algorithme $\mathcal{D}$ est appelé algorithme de déchiffrement. C'est un algorithme déterministe qui prend en entrée une clef secrète* sk *et un chiffré c élément de $\mathcal{C}$ et qui retourne la valeur $\mathcal{D}_{\text{sk}}(c)$. Cette valeur sera soit un message clair élément de $\mathcal{M}$, soit une erreur notée $\perp$.*

Le triplet $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ doit également vérifier la propriété suivante : pour tout couple (pk, sk) *retourné par l'algorithme de génération $\mathcal{K}$, pour tout message m appartenant à $\mathcal{M}$, et pour tout c retourné par $\mathcal{E}_{\text{pk}}(m)$, on doit avoir $\mathcal{D}_{\text{sk}}(c) = m$. Enfin, les trois algorithmes $\mathcal{K}$, $\mathcal{E}$, et $\mathcal{D}$ doivent s'exécuter en temps polynomial.*

Pour que le schéma soit intéressant, il faut qu'il soit difficile de déchiffrer un message sans la connaissance de la clef secrète, *i.e.*, que le schéma soit sûr.

Une manière de construire un schéma de chiffrement asymétrique probabiliste sûr sera d'avoir une famille de fonctions trappe probabilistes à sens-unique. C'est à dire une famille de fonctions facilement évaluables (*i.e.*, en temps polynomial), mais difficiles à inverser ponctuellement, *i.e.*, pour chaque fonction, étant donnée une image il sera difficile de retrouver l'antécédent correspondant. C'est la notion de sens-unique. Cette notion de sécurité sera abordée plus formellement dans la section suivante.

Le fait que ces fonctions soient à trappe signifie que pour chacune de ces fonctions celui qui connaît une certaine information, la trappe, dispose d'un algorithme polynomial pour inverser la fonction, c'est à dire retrouver l'antécédent de chaque image.

Dans cette construction, l'algorithme de génération de clef retournera comme clef publique une fonction f de la famille de fonctions trappe probabilistes à sens-unique, et comme clef privée, la trappe t correspondante. L'algorithme de chiffrement sera alors la fonction f, et l'algorithme de déchiffrement, l'algorithme d'inversion de la fonction f au moyen de l'information t.

C'est cette construction que l'on utilisera implicitement dans le chapitre IV à partir des fonctions trappe probabilistes introduites au chapitre II.

Propriétés

Pour déterminer l'efficacité d'un système, on calculera le coût algorithmique des opérations de chiffrement et de déchiffrement. On s'intéressera également à un autre paramètre, l'expansion.

Comme la fonction de chiffrement est probabiliste, l'espace des messages chiffrés $\mathcal{C}$ sera de cardinal plus grand que celui des messages clairs $\mathcal{M}$ (on suppose que ces ensembles sont finis). Par conséquent, on aura besoin de plus de bits pour coder les chiffrés que pour coder les messages clairs. On définit l'expansion dans la définition suivante.

Définition I – 2. *Soit $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ un schéma de chiffrement asymétrique probabiliste. Étant donné un paramètre de sécurité k et* (pk, sk) *retourné par l'algorithme de génération de clefs, l'expansion du système sera le rapport de la taille maximale (en bits) de la sortie de l'algorithme $\mathcal{E}_{\text{pk}}$ sur la taille maximale des messages clairs.*

Dans la pratique, on souhaitera avoir un expansion la plus proche possible de 1.

Certains schémas que l'on va étudier satisferont une propriété supplémentaire donnée dans la définition suivante.

Définition I – 3. *Soit* $\Pi = (\mathscr{K}, \mathscr{E}, \mathscr{D})$ *un schéma de chiffrement asymétrique probabiliste. Le système sera dit homomorphique si, étant donné un paramètre de sécurité* k *et* $(\mathrm{pk}, \mathrm{sk})$ *retourné par l'algorithme de génération de clef, l'ensemble des messages* $\mathscr{M}$ *(resp. l'ensemble des chiffrés* $\mathscr{C}$*) est muni d'une loi de groupe notée additivement (resp. multiplicativement) telle qu'étant donnés deux messages* m_1 *et* m_2*,* $c_1 \leftarrow \mathscr{E}_{\mathrm{pk}}(m_1)$ *et* $c_2 \leftarrow \mathscr{E}_{\mathrm{pk}}(m_2)$ *alors* $c_1 c_2$ *est une sortie possible de* $\mathscr{E}_{\mathrm{pk}}(m_1 + m_2)$*.*

Remarquons que si $\mathscr{E}_{\mathrm{pk}}$ réalise une surjection de $\mathscr{M}$ sur $\mathscr{C}$ alors le schéma sera homomorphique si et seulement si l'algorithme de déchiffrement $\mathscr{D}_{\mathrm{sk}}$ est un morphisme de groupes.

De nombreux cryptosystèmes probabilistes sont homomorphiques : le système fondateur de la cryptographie probabiliste, le système de Golwasser-Micali (cf. [GM84] et page 80), ou le système El Gamal, pour ne citer que les plus connus. On construira, en sous-section II–1.2, une famille de fonctions trappe probabilistes qui permettra de créer d'autres cryptosystèmes probabilistes homomorphiques en section IV–2.

Cette propriété permet de nombreuses applications. Par exemple, un schéma homomorphique sera un ingrédient essentiel de certaines constructions de schémas de vote électronique : étant donnés les chiffrés de chaque vote, on pourra obtenir le résultat du vote par déchiffrement du produit de ces chiffrés sans devoir déchiffrer les votes individuellement. On réfère le lecteur à [CGS97, HS00] pour les détails d'une telle construction. Certains systèmes étudiés au chapitre IV ont été utilisés pour construire de tels schémas de vote : le schéma de Damgård-Jurick exposé page 86 (cf. [DJ01]) et le schéma de Benaloh exposé page 81 (cf. [Ben88]).

2. Notions de sécurité

Sécurité

Comme vu dans la section précédente, la notion de sécurité la plus naturelle pour un schéma de chiffrement probabiliste Π est celle de sens-unique. De manière informelle, un attaquant ne doit pas pouvoir, étant donné une clef publique pk et un chiffré retourné par $\mathscr{E}_{\mathrm{pk}}(m)$, retrouver m en temps polynomial. Les définitions suivantes permettent de formaliser les choses.

Définition I – 4. *Une fonction* $\varepsilon : \mathbf{N} \to \mathbf{R}$ *est négligeable si pour toute constante* $c \geqslant 0$*, il existe un entier* k_c *tel que* $\varepsilon(k) \leqslant k^{-c}$ *pour tout* $k \geqslant k_c$*.*

Définition I – 5 (sens-unique)**.** *Soit* $\Pi = (\mathscr{K}, \mathscr{E}, \mathscr{D})$ *un schéma de chiffrement asymétrique probabiliste. Soit* $\mathscr{A}$ *un algorithme attaquant* Π*. On définit sa probabilité de succès dans l'inversion ponctuelle du schéma par*

$$\mathrm{Succ}^{\mathrm{OW}}_{\Pi,\mathscr{A}}(k) = \mathrm{P}\left[(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathscr{K}\left(1^k\right),\ c \leftarrow \mathscr{E}_{\mathrm{pk}}(m),\ m \leftarrow \mathscr{A}(\mathrm{pk}, c)\right].$$

On dira que le schéma Π *est sûr si la fonction* $k \mapsto \mathrm{Succ}^{\mathrm{OW}}_{\Pi,\mathscr{A}}(k)$ *est négligeable dès que* $\mathscr{A}$ *est un algorithme polynomial.*

Sécurité sémantique

Une seconde notion plus forte est qu'un attaquant ne puisse extraire aucune information en temps polynomial sur un message clair à partir de l'un de ces chiffrés, en dehors de celles qu'il aurait pu obtenir sans ce chiffré. C'est la notion de sécurité sémantique introduite dans [GM84]. Dans la pratique, on utilise une notion plus simple à manipuler, celle d'indistinguabilité : si m_1 et m_2 sont deux messages clairs et si c est un chiffré de l'un de ces deux messages, alors un attaquant ne peut pas décider si c a été retourné par $\mathcal{E}_{\mathrm{pk}}(m_1)$ ou par $\mathcal{E}_{\mathrm{pk}}(m_2)$, toujours en temps polynomial. Cette notion, introduite aussi par Goldwasser et Micali dans [GM84], a été prouvée équivalente à la notion de sécurité sémantique dans [MRS88].

Pour donner une définition formelle de cette notion il faut tout d'abord définir les moyens d'un attaquant. On considère une attaque à messages clairs choisis (notée CPA, pour *chosen-plaintext attack*). L'attaque de chaque notion se passe alors en deux phases. Dans la première, l'attaquant, au vu de la clef publique, choisit des messages pour lesquels il estime pouvoir attaquer la notion. Dans la seconde, il essaye de résoudre un défi, constitué d'un chiffré obtenu en fonction des messages choisis.

On donne maintenant la définition formelle de l'indistinguabilité.

Définition I – 6 (IND − CPA)**.** *On définit un algorithme $\mathcal{A} := (\mathcal{A}_1, \mathcal{A}_2)$ attaquant l'indistinguabilité d'un schéma $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. Étant donné une clef publique* pk*, l'algorithme $\mathcal{A}_1$ sort un triplet (m_0, m_1, s) contenant deux messages de même longueur et un état d'information s. Le défi est un chiffré c de m_b où b est un bit tiré au hasard. L'algorithme $\mathcal{A}_2$ prend en entrée (m_0, m_1, s, c) et doit donner la valeur de b.*

Soit k un paramètre de sécurité, l'avantage de l'attaquant $\mathcal{A}$ pour résoudre l'indistinguabilité du schéma Π est défini par

$$\mathrm{Adv}^{\mathrm{IND}}_{\Pi,\mathcal{A}}(k) = \left| 2 \times \mathrm{P}\left[\begin{array}{l} (\mathrm{pk}, \mathrm{sk}) \leftarrow \mathcal{K}\left(1^k\right),\ (m_0, m_1, s) \leftarrow \mathcal{A}_1(\mathrm{pk}), \\ b \leftarrow \{0,1\},\ c \leftarrow \mathcal{E}_{\mathrm{pk}}(m_b),\ b' \leftarrow \mathcal{A}_2(m_0, m_1, s, c) \ :\ b' = b \end{array} \right] - 1 \right|.$$

Le schéma Π est sûr au sens IND − CPA *si la fonction $k \mapsto \mathrm{Adv}^{\mathrm{IND}}_{\Pi,\mathcal{A}}(k)$ est négligeable dès que $\mathcal{A}$ est un algorithme polynomial.*

Remarque. Dans la définition précédente, le terme avantage signifie l'avantage de l'algorithme $\mathcal{A}$ par rapport à un lancer de pièce : si $\mathcal{A}_2$ tire au hasard sa réponse, on aura $\mathrm{Adv}^{\mathrm{IND}}_{\Pi,\mathcal{A}}(k) = 0$.

En utilisant les formules

$$\begin{aligned} \mathrm{P}[b' = b] &= \mathrm{P}[b' = b | b = 1] \times \mathrm{P}[b = 1] + \mathrm{P}[b' = b | b = 0] \times \mathrm{P}[b = 0] \\ &= \frac{\mathrm{P}[b' = 1 | b = 1]}{2} + \frac{\mathrm{P}[b' = 0 | b = 0]}{2}, \end{aligned}$$

et

$$\mathrm{P}\left[b' = 1 | b = 0\right] + \mathrm{P}\left[b' = 0 | b = 0\right] = 1,$$

on voit facilement qu'avec les notations de la définition, on a également les expressions suivantes

$$\begin{aligned} \mathrm{Adv}^{\mathrm{IND}}_{\Pi,\mathscr{A}}(k) &= \Big|\, \mathrm{P}[b'=1|b=1] - \mathrm{P}[b'=1|b=0] \,\Big| \\ &= \Big|\, \mathrm{P}[b'=0|b=1] - \mathrm{P}[b'=0|b=0] \,\Big|. \end{aligned}$$

Pour les systèmes que l'on va introduire dans le chapitre II une notion sera plus adaptée, celle, intuitive, de non-reconnaissance de chiffrés : on ne veut pas qu'un attaquant puisse dire si un chiffré donné correspond à un message donné. Cette notion est à rapprocher de la notion "real or random" introduite et étudiée dans le cadre de la cryptographie symétrique par Bellare, Desai *et al.* dans [BDJR97]. On montre dans la suite que cette notion est équivalente à celle de l'indistinguabilité.

On donne d'abord la définition formelle de la non-reconnaissance de chiffrés.

Définition I – 7 (REC – CPA)**.** *On définit un algorithme $\mathscr{B} := (\mathscr{B}_1, \mathscr{B}_2)$ attaquant la non-reconnaissance de chiffrés d'un schéma $\Pi = (\mathscr{K}, \mathscr{E}, \mathscr{D})$. Étant donné une clef publique* pk*, l'algorithme $\mathscr{B}_1$ sort un couple (m, s) contenant un message et un état d'information s. On tire un bit b au hasard. Le défi est un chiffré c de m si b vaut 1 et un chiffré d'un message aléatoire de même longueur si b est nul. L'algorithme $\mathscr{B}_2$ prend en entrée (m, s, c) et doit donner la valeur de b,* i.e.*, dire si c est un chiffré de m.*

Soit k un paramètre de sécurité, l'avantage de l'attaquant $\mathscr{B}$ pour résoudre le problème de reconnaissance des chiffrés du schéma Π est défini par

$$\mathrm{Adv}^{\mathrm{REC}}_{\Pi,\mathscr{B}}(k) = \left| 2 \times \mathrm{P}\left[\begin{array}{l} (\mathrm{pk}, \mathrm{sk}) \leftarrow \mathscr{K}\left(1^k\right), (m, s) \leftarrow \mathscr{B}_1(\mathrm{pk}), b \leftarrow \{0,1\}, m' \leftarrow \mathscr{M}, \\ c_0 \leftarrow \mathscr{E}_{\mathrm{pk}}(m'),\ c_1 \leftarrow \mathscr{E}_{\mathrm{pk}}(m),\ b' \leftarrow \mathscr{B}_2(m, s, c_b) \,:\, b' = b \end{array} \right] - 1 \right|.$$

Le schéma Π est sûr au sens REC – CPA *si la fonction $k \mapsto \mathrm{Adv}^{\mathrm{IND}}_{\Pi,\mathscr{B}}(k)$ est négligeable dès que $\mathscr{B}$ est un algorithme polynomial.*

Le théorème suivant n'est jamais clairement énoncé dans la littérature concernant la cryptographie asymétrique. Les éléments de la preuve sont pourtant très souvent utilisés pour analyser la sécurité sémantique de nouveaux schémas. Ainsi, la construction de l'algorithme $\mathscr{B}$ dans la seconde partie de la preuve, est similaire à celle faite par Pointcheval dans [Poi99] et par Catalano *et al.* dans [CGHGN01] pour étudier la sécurité sémantique de leurs systèmes respectifs. On retrouve également cette construction lors de l'analyse de la sécurité sémantique du cryptosystème ElGamal.

Théorème I – 8. *Les notions* REC – CPA *et* IND – CPA *sont équivalentes.*

Démonstration. Soit $\Pi = (\mathscr{K}, \mathscr{E}, \mathscr{D})$ un schéma. On suppose que Π est sûr au sens IND – CPA et on montre qu'il est sûr au sens REC – CPA. Pour cela, on se donne un paramètre de sécurité k et une clef publique pk retournée par $\mathscr{K}(1^k)$. On considère un attaquant $\mathscr{B} := (\mathscr{B}_1, \mathscr{B}_2)$ cherchant à reconnaître les chiffrés de Π et on montre que son avantage est faible. On construit un algorithme $\mathscr{A} := (\mathscr{A}_1, \mathscr{A}_2)$ attaquant l'indistinguabilité de Π au moyen d'appels à $\mathscr{B}$ comme suit :

$\mathscr{A}_1(\mathrm{pk})$	$\mathscr{A}_2(m_0, m_1, s, c)$
$(m_0, s) \leftarrow \mathscr{B}_1(\mathrm{pk})$ $m_1 \leftarrow \mathscr{M}$ **sortie** : (m_0, m_1, s)	$b' \leftarrow \mathscr{B}_2(m_0, s, c)$ **sortie** : $\overline{b'}$

On note b le bit tel que $c = \mathscr{E}(m_b)$. Si $b = 0$, c est un chiffré de m_0 et sinon, c est un chiffré d'un message aléatoire pour $\mathscr{B}_2$. D'après la remarque page 10, on a

$$\begin{aligned}
\mathrm{Adv}^{\mathrm{IND}}_{\Pi,\mathscr{A}}(k) &= \left| \mathrm{P}\left[\overline{b'} = 1 | b = 1\right] - \mathrm{P}\left[\overline{b'} = 1 | b = 0\right] \right| \\
&= \left| \mathrm{P}[b' = 0 | b = 1] - \mathrm{P}[b' = 0 | b = 0] \right| \\
&= \mathrm{Adv}^{\mathrm{REC}}_{\Pi,\mathscr{B}}(k).
\end{aligned}$$

Ainsi, comme Π est sûr au sens $\mathrm{IND}-\mathrm{CPA}$, l'avantage de $\mathscr{A}$ est faible et donc celui de $\mathscr{B}$ aussi. On a montré que Π est sûr au sens $\mathrm{REC}-\mathrm{CPA}$.

Réciproquement, on suppose que Π est sûr au sens $\mathrm{REC}-\mathrm{CPA}$. On se donne un paramètre de sécurité k et une clef publique pk retournée par $\mathscr{K}(1^k)$. On considère un algorithme $\mathscr{A} := (\mathscr{A}_1, \mathscr{A}_2)$ attaquant l'indistinguabilité de Π. On construit un algorithme $\mathscr{B} := (\mathscr{B}_1, \mathscr{B}_2)$ cherchant à reconnaître les chiffrés de Π comme suit :

$\mathscr{B}_1(\mathrm{pk})$	$\mathscr{B}_2(m_d, s', c)$ où $s' = (m_{\overline{d}}, d, s)$
$(m_0, m_1, s) \leftarrow \mathscr{A}_1(\mathrm{pk})$ $d \leftarrow \{0,1\}$ $s' \leftarrow (m_{\overline{d}}, d, s)$ **sortie** : (m_d, s')	$b' \leftarrow \mathscr{A}_2(m_0, m_1, s, c)$ **si** $d = b'$ **alors** $\quad e := 1$ **sinon** $\quad e := 0$ **sortie** : e

On note b le bit correspondant au défi lancé à $\mathscr{B}$. Supposons que b soit égal à 1. Le chiffré c est alors un chiffré de m_d. L'algorithme $\mathscr{B}_2$ va sortir un 1 si $d = b'$, *i.e.*, si l'algorithme $\mathscr{A}_2$ devine correctement le bit d. On a donc,

$$\mathrm{P}\left[e = 1 | b = 1\right] = \mathrm{P}\left[\begin{array}{l} (m_0, m_1, s) \leftarrow \mathscr{A}_1(\mathrm{pk}),\ d \leftarrow \{0,1\}, \\ c \leftarrow \mathscr{E}_{\mathrm{pk}}(m_d),\ b' \leftarrow \mathscr{A}_2(m_0, m_1, s, c) \ :\ b' = d \end{array}\right].$$

Par définition, cette dernière probabilité vaut

$$\frac{1}{2} \pm \frac{\mathrm{Adv}^{\mathrm{IND}}_{\Pi,\mathscr{A}}(k)}{2}.$$

Quand b vaut 0, c est un chiffré d'un message aléatoire. Il est donc indépendant de la valeur de d, l'algorithme $\mathscr{A}_2$ ne peut alors que retourner la valeur 0 ou 1 avec une probabilité $\frac{1}{2}$.

On a donc

$$\begin{aligned}\mathrm{Adv}^{\mathrm{REC}}_{\Pi,\mathscr{B}}(k) &= \left|\, \mathrm{P}[e=1|b=1] - \mathrm{P}[e=1|b=0] \,\right| \\ &= \left| \frac{1}{2} \pm \frac{\mathrm{Adv}^{\mathrm{IND}}_{\Pi,\mathscr{A}}(k)}{2} - \frac{1}{2} \right| \\ &= \frac{\mathrm{Adv}^{\mathrm{IND}}_{\Pi,\mathscr{A}}(k)}{2}.\end{aligned}$$

Comme on a supposé Π sûr au sens REC − CPA, l'avantage de $\mathscr{B}$ est faible, donc celui de $\mathscr{A}$ contre l'indistinguabilité l'est aussi, donc Π est sûr au sens IND − CPA. □

Moyens de l'attaquant

Comme précédemment on se placera systématiquement, dans la suite, dans le cadre d'une attaque à message clairs choisis inévitable dans le cadre de systèmes à clefs publiques.

Dans un modèle de sécurité supérieur, on autorise aussi les attaques à chiffrés choisis (notées CCA pour *chosen-ciphertext-attack*). Dans ce cadre, un attaquant a accès à un oracle de déchiffrement. Si cet accès n'est possible qu'avant un défi on parle d'attaques non adaptatives (CCA1), si cet accès est illimité (sans pouvoir demander le déchiffrement du défi), on parle d'attaques adaptatives (CCA2).

Les schémas homomorphiques ne peuvent être sûr dans le cadre d'attaques CCA2. En effet, considérons un schéma de chiffrement asymétrique probabiliste homomorphique $\Pi = (\mathscr{K}, \mathscr{E}, \mathscr{D})$. Soit k un paramètre de sécurité et $(\mathrm{pk}, \mathrm{sk})$ retourné par l'algorithme de génération de clef Π. À partir d'un défi c_1 chiffré d'un message m_1, on peut produire un nouveau chiffré relié à c_1 : le chiffré $c_1 c_2$ où $c_2 \leftarrow \mathscr{E}_{\mathrm{pk}}(m_1)$ avec m_2 un message tiré au hasard. On présente alors $c_1 c_2$ à l'oracle de déchiffrement qui retourne un message m_3. On a alors $m_1 = m_3 - m_2$.

Les systèmes non homomorphiques découlant de la fonction trappe présentée en sous-section II–2.1 souffriront aussi de la même attaque dans le modèle de sécurité CCA2. Notons que l'on peut cependant leur appliquer une construction décrite par Pointcheval et Paillier dans [PP99] pour les rendre sûr, dans le modèle de l'oracle aléatoire, face à de telles attaques.

CHAPITRE II

FONCTIONS TRAPPE PROBABILISTES

Dans ce chapitre, on introduit plusieurs fonctions trappe probabilistes. Ces fonctions sont introduites de façon générique, *i.e.*, on n'explicite pas les ensembles de départs et d'arrivées. Ce sont essentiellement des généralisations de cryptosystèmes existants (exceptée la fonction trappe présentée en sous-section 2.2). On montrera dans le chapitre IV dans quelle mesure les systèmes existants dérivent de ces généralisations et comment utiliser les fonctions trappe introduites ici pour produire de nouveaux schémas effectifs.

1. Fonctions trappe probabilistes homomorphiques

Les résultats de cette section généralisent, dans un groupe muni des bonnes hypothèses, ceux notamment obtenus par Benaloh dans $(\mathbf{Z}/n\mathbf{Z})^\times$ (cf. [Ben88] et page 81) et par Paillier dans $\left(\mathbf{Z}/n^2\mathbf{Z}\right)^\times$ (cf. [Pai99] et page 84) avec n un entier RSA.

Notations

Dans cette section, on notera G un groupe abélien fini multiplicatif. On se donne un entier k et on note G^k le sous-groupe de G constitué par les puissances k–ièmes, *i.e.*,

$$\mathrm{G}^k = \left\{ x \in \mathrm{G}, \exists\, y \in \mathrm{G}, x = y^k \right\}.$$

On supposera dans la suite que $k \,\big|\, |\mathrm{G}|$, et on notera $\lambda := |\mathrm{G}|\,/k$. De plus, on supposera que λ et k sont premiers entre eux.

On donne d'abord quelques résultats permettant d'établir la structure du groupe quotient G/G^k puis on voit comment utiliser ce groupe quotient afin de construire un système probabiliste homomorphique dont la sécurité sémantique est équivalente à la reconnaissance des puissances k–ièmes de G.

1.1. Groupes abéliens finis multiplicatifs et puissances k–ièmes

On commence par donner un théorème, fondamental pour la suite, découlant des hypothèses prises sur l'entier k.

Théorème II – 1. *Soient* G *un groupe abélien fini multiplicatif et* k *un entier divisant* $|\mathrm{G}|$ *tel que* $|\mathrm{G}|$ *et* $|\mathrm{G}|/k$ *soient premiers entre eux. Tout élément de* G^k *a alors exactement* k *racines* k*–ièmes.*

Démonstration. Soit $a \in \mathrm{G}^k$, on cherche le nombre de racines k–ièmes de a. Le groupe G est isomorphe à une somme directe de ℓ groupes cycliques G_i avec $i \in \{1,\ldots,\ell\}$. Dans chaque composante cyclique G_i, l'équation $a = x^k$ donne au moins une solution puisque a appartient à G^k. On obtient donc

$$k_i := \mathrm{pgcd}(k, |\mathrm{G}_i|)$$

solutions dans chaque G_i, pour i dans $\{1,\ldots,\ell\}$ (c'est le nombre de racines k–ièmes de l'unité dans chaque groupe cyclique G_i).

Dans G, on a donc $\prod_{i=1}^{\ell} k_i$ racines k–ièmes. Calculons ce nombre. Pour cela, on écrit la décomposition de $|\mathrm{G}|$ en produit de nombres premiers p_j distincts :

$$|\mathrm{G}| = \prod_{j \in \mathrm{I}} p_j^{\alpha_j},$$

où les α_j, avec j élément de I, sont tous des entiers naturels non nuls.

Comme $k \mid |\mathrm{G}|$ et $\mathrm{pgcd}(|\mathrm{G}|, |\mathrm{G}|/k) = 1$, la décomposition de k en produit de nombres premiers distincts est

$$k = \prod_{j \in \mathrm{J} \subseteq \mathrm{I}} p_j^{\alpha_j},$$

pour un certain sous-ensemble J de I.

Comme $\prod_{1 \leqslant i \leqslant \ell} |\mathrm{G}_i| = |\mathrm{G}|$, chaque ordre $|\mathrm{G}_i|$ se décompose en produit de nombres premiers distincts :

$$|\mathrm{G}_i| = \prod_{j \in \mathrm{I}} p_j^{\beta_{i,j}},$$

les $\beta_{i,j}$, avec j dans I et i dans $\{1,\ldots,\ell\}$, étant des entiers naturels éventuellement nuls tels que pour tout $j \in \mathrm{I}$,

$$\sum_{1 \leqslant i \leqslant \ell} \beta_{i,j} = \alpha_j.$$

Comme pour tout $j \in \mathrm{I}$, et tout $i \in \{1,\ldots,\ell\}$, $\beta_{i,j} \leqslant \alpha_j$, la décomposition de $k_i = \mathrm{pgcd}(k, |\mathrm{G}_i|)$ est

$$k_i = \prod_{j \in \mathrm{J}} p_j^{\beta_{i,j}},$$

on obtient finalement

$$\prod_{i=1}^{\ell} k_i = \prod_{i=1}^{\ell} \prod_{j \in \mathrm{J}} p_j^{\beta_{i,j}} = \prod_{j \in \mathrm{J}} p_j^{\sum_{i=1}^{\ell} \beta_{i,j}} = \prod_{j \in \mathrm{J}} p_j^{\alpha^j} = k.$$

□

Corollaire II – 2. *Le groupe quotient* G/G^k *est d'ordre* k.

Démonstration. On a vu que chaque élément de G^k a k racines k–ièmes, en particulier, on a k racines k–ièmes de l'unité. Ainsi, si on note f le morphisme : $G \to G, x \mapsto x^k$, le noyau de f est d'ordre k. Comme $\mathrm{Im}\, f = G^k$, on en déduit que G^k est d'ordre λ et que le groupe quotient G/G^k est d'ordre k. □

On montre maintenant des lemmes qui nous permettront d'utiliser le sous-groupe G^k et le quotient G/G^k.

Lemme II – 3. *On a la caractérisation :*

$$G^k = \left\{ x \in G,\ x^\lambda = 1 \right\}.$$

Démonstration. Si $x \in G^k$, on a $x = y^k$ pour un élément y de G, donc $x^\lambda = y^{|G|} = 1$. Réciproquement, on suppose que $x^\lambda = 1$. Comme $\mathrm{pgcd}(k, \lambda) = 1$, la relation de Bézout assure l'existence d'entiers A et B tels que $Ak + B\lambda = 1$. On a donc la relation

$$x = x^{Ak} x^{B\lambda} = \left(x^A\right)^k,$$

qui montre que x est bien une puissance k–ième. □

Remarque. Une généralisation immédiate du lemme II – 3 consiste à utiliser un exposant quelconque du groupe G au lieu de l'entier $|G|$, *i.e.*, un entier Λ tel que pour tout élément x de G, $x^\Lambda = 1$. Il faut alors redéfinir λ en conséquence : on pose $\lambda := \Lambda/k$ en supposant toujours que k divise Λ et que $\mathrm{pgcd}(\lambda, k) = 1$.

Lemme II – 4. *Si g est un élément de* G *d'ordre divisible par k alors $\overline{g}$ est un générateur de* G/G^k, *en désignant par $\overline{g}$ la classe de g modulo* G^k.

Démonstration. Soit g un élément de G d'ordre k. On note ℓ l'ordre de $\overline{g}$ dans G/G^k. Comme

$$\left(g^k\right)^\lambda = g^{|G|} = 1,$$

le lemme précédent assure que ℓ divise k.

Soit m un entier tel que l'ordre de g dans G soit mk. L'ordre de g^λ est alors

$$\frac{mk}{\mathrm{pgcd}(mk, \lambda)}.$$

Comme $|G| = \lambda k$, avec $\mathrm{pgcd}(k, \lambda) = 1$ et $(mk) \mid |G|$, on a $m \mid \lambda$ et $\mathrm{pgcd}(mk, \lambda) = m$. On en déduit que l'ordre de g^λ est k.

Par définition de ℓ, on a $g^\ell \in G^k$, donc, d'après le lemme II – 4, on a $g^{\ell\lambda} = 1$. On en déduit que l'ordre de g^λ divise ℓ, *i.e.*, que k divise ℓ. Ainsi, on a montré que $\ell = k$, et donc que $\overline{g}$ engendre G/G^k. □

Remarque. En intervertissant le rôle de λ et de k, le lemme II – 3 donne

$$G^\lambda = \left\{ x \in G, x^k = 1 \right\}.$$

Autrement dit, G^λ est le sous-groupe des racines k–ièmes de l'unité de G. D'autre part, ce même lemme montre que le noyau du morphisme : $G \to G, x \mapsto x^\lambda$ est G^k. Ainsi, on a l'isomorphisme

$$G/G^k \xrightarrow{\sim} G^\lambda.$$

Si le groupe quotient G/G^k est cyclique, G^λ le sera aussi et réciproquement. Pour trouver un générateur éventuel de G/G^k et satisfaire aux hypothèses du lemme II – 4, on se limitera à chercher un élément d'ordre k. On cherchera donc cet élément parmi les générateurs éventuels de G^λ, *i.e.*, du sous-groupe d'ordre k constitué par les racines k–ièmes de l'unité de G.

1.2. Une fonction trappe basée sur le groupe quotient G/G^k

On veut construire un système à clef publique probabiliste et homomorphique dont la sécurité sémantique est basée sur la reconnaissance des éléments de G^k dans G. Cette construction généralisera le cryptosystème de Paillier exposé en page 84. Ce système utilise $G = (\mathbf{Z}/n^2\mathbf{Z})^\times$ où n est un entier RSA et $k = n$.

On verra dans la sous-section IV–2 que d'autres cryptosystèmes existants peuvent être considérés comme des « instanciations » de cette fonction trappe générique. On verra également comment en construire d'autres (cf. page 92), en prenant pour groupe G un groupe issu des quotients de corps quadratiques, défini en sous-section III–3.1.

Données publiques

On suppose publics le groupe G, l'entier k et un élément g de G d'ordre k[1]. Les messages à chiffrer seront des éléments de $\mathbf{Z}/k\mathbf{Z}$. On suppose connu un générateur aléatoire d'éléments de G^k ainsi qu'un algorithme de calcul du logarithme discret dans $\langle g \rangle$, *i.e.*, un algorithme qui étant donné $c \in \langle g \rangle$, retourne l'élément m de $\mathbf{Z}/k\mathbf{Z}$ tel que $c = g^m$.

Prototype de l'algorithme de chiffrement

$$\mathcal{E}_{G,k,g} : \begin{cases} \mathbf{Z}/k\mathbf{Z} & \longrightarrow & G \\ m & \longmapsto & g^m \rho \end{cases}$$

où ρ est un élément aléatoire de G^k, obtenu avec distribution uniforme.

D'après le lemme II – 4, la classe $\overline{g}$ engendre le groupe quotient G/G^k cyclique d'ordre k. Ainsi, si $c \leftarrow \mathcal{E}_{G,k,g}(m)$, m est le logarithme discret de $\overline{c}$ en base $\overline{g}$ dans G/G^k. On notera, dans $\mathbf{Z}/k\mathbf{Z}$,

$$[\![c]\!]_g := \log_{\overline{g}}(\overline{c}).$$

La fonction de déchiffrement associée à $\mathcal{E}_{G,k,g}$ sera donc un morphisme surjectif de $(G, \times)$ dans $(\mathbf{Z}/k\mathbf{Z}, +)$. Par conséquent, un cryptosystème basé sur la primitive $\mathcal{E}_{G,k,g}$ sera homomorphique.

(1). En fait, on peut facilement adapter ce qui suit pour utiliser un élément d'ordre un multiple de k, on perd cependant certaines propriétés notamment l'homomorphie de la fonction $\mathcal{E}'_{G,k,g}$ (cf. page 22).

Clef privée et déchiffrement

La trappe permettant d'inverser la fonction $\mathscr{E}_{G,k,g}$ est l'entier λ dont la connaissance est équivalente à celle de l'ordre du groupe G. Notons que l'entier λ permet de travailler dans le groupe quotient G/G^k. En effet, pour deux éléments x et y de G, on aura $\overline{x} = \overline{y}$ si et seulement si $x^\lambda = y^\lambda$.

Soit c un élément de G à déchiffrer, *i.e.*, on doit retrouver m dans $\mathbf{Z}/k\mathbf{Z}$ tel qu'il existe ρ dans G^k avec $c = g^m\rho$. On a

$$c^\lambda = g^{m\lambda}.$$

Ainsi, on se ramène à un calcul de logarithme discret dans $\langle g \rangle$ pour lequel on dispose d'un algorithme public. On récupère $m\lambda$ dans l'anneau $\mathbf{Z}/k\mathbf{Z}$, puis m car λ et k sont premiers entre eux.

Sécurité

La sécurité du système repose sur la difficulté du problème suivant : étant donné c un élément quelconque de G, trouver m dans $\mathbf{Z}/k\mathbf{Z}$ tel que $m = [\![c]\!]_g$. Ce problème est auto-réductible aléatoirement (*random-self-reducible*), *i.e.*, une instance c_1 du problème peut être transformée en une instance aléatoire c_2 avec distribution uniforme. Pour cela, on pose $c_2 = c_1 g^m \rho$, en tirant aléatoirement m dans $\mathbf{Z}/k\mathbf{Z}$ et ρ dans G^k. Si on sait trouver $[\![c_2]\!]_g$, on aura

$$[\![c_1]\!]_g = [\![c_2]\!]_g - m.$$

Toutes les instances sont donc de difficultés équivalentes.

D'autre part, du fait des propriétés du logarithme discret, ce problème ne dépend pas du choix de g, en effet si g_1 et g_2 sont deux éléments de G d'ordre k, alors pour tout élément c de G,

$$[\![c]\!]_{g_1} = [\![c]\!]_{g_2} [\![g_2]\!]_{g_1}.$$

Cette égalité appliquée dans le cas $c = g_1$, permet d'établir que $[\![g_2]\!]_{g_1}$ est inversible dans l'anneau $\mathbf{Z}/k\mathbf{Z}$. Deux appels à un oracle résolvant $[\![.]\!]_{g_1}$ permettent donc de retrouver $[\![c]\!]_{g_2}$, et réciproquement comme g_1 et g_2 jouent des rôles symétriques.

Maintenant que l'on a vu que le problème du déchiffrement ne dépend pas de g, on donne la définition formelle du problème sur lequel repose la sécurité du cryptosystème.

Définition II – 5. *On notera* $\text{Classe}_{G,k}$ *et on appellera problème de classe de résidualité d'ordre* k *dans* G *le problème suivant : Étant donnés* c *un élément de* G *et un élément* g *de* G *d'ordre* k*, calculer* $[\![c]\!]_g$.

D'après la discussion qui précède sur le déchiffrement du cryptosystème, on a la réduction suivante.

Théorème II – 6. *Soient* G *un groupe abélien fini multiplicatif et* k *un entier divisant* $|G|$ *tel que* $|G|$ *et* $|G|/k$ *soient premiers entre eux. On note* Ordre_G *le problème consistant à retrouver*

l'ordre de G *et* $\mathrm{Dlog}_{\langle g\rangle}$ *le problème du logarithme discret dans* $\langle g\rangle$ *étant donné un élément* g *de* G *d'ordre* k*. On a*

$$\mathrm{Classe}_{G,k} \overset{\mathscr{P}}{\Longleftarrow} \left(\mathrm{Ordre}_G \wedge \mathrm{Dlog}_{\langle g\rangle}\right).$$

Remarque. Dans la pratique, comme on suppose $\mathrm{Dlog}_{\langle g\rangle}$ facile, la connaissance seule de l'ordre λk du groupe G permet de casser le système, le système sera donc faible si λ est petit.

On étudie maintenant les relations entre $\mathrm{Classe}_{G,k}$ et un autre problème. Pour cela, on regarde le problème du déchiffrement sous un autre angle. Au lieu de passer au quotient dans G/G^k pour éliminer la puissance k–ième puis retrouver le message par un problème de logarithme discret, on va passer au quotient dans $G/\langle g\rangle$ pour récupérer en premier lieu la puissance k–ième. On en déduira ensuite le message, toujours par le calcul d'un logarithme discret dans $\langle g\rangle$.

On a vu dans la remarque page 17 que le noyau du morphisme de $f : G \to G, x \mapsto x^k$ est G^λ. Notons que g est un élément de G^λ. Comme G^λ et g sont d'ordre k, on a en fait l'égalité : $G^\lambda = \langle g\rangle$. Par passage au quotient, le morphisme f donne l'isomorphisme :

$$G/G^\lambda \xrightarrow{\sim} G^k,$$

ainsi en regardant les éléments de G modulo G^λ, on a une correspondance explicite avec les puissances k–ièmes de G. Comme k est premier avec λ et comme G/G^λ est d'ordre λ, le morphisme

$$G/G^\lambda \longrightarrow G/G^\lambda, x \longmapsto x^k$$

est un automorphisme de G/G^λ. Ceci conduit à la définition d'un nouveau problème :

Définition II – 7.

Par analogie avec le système RSA, on note $\mathrm{RSA}_{G/G^\lambda,k}$ *et on appelle problème RSA d'ordre* k *dans* G/G^λ *le problème d'inversion ponctuelle de l'automorphisme* $x \mapsto x^k$ *de* G/G^λ, i.e., *étant donné* c *un élément de* G/G^λ*, retrouver* x *dans* G/G^λ *tel que* $x^k = c$.

On note $\mathrm{C\text{–}RSA}_{G,k}$ *et on appelle problème de classe RSA d'ordre* k *dans* G*, le problème suivant : étant donné un élément* c *de* G*, trouver* x *dans* G *tel que*

$$x^k \equiv c \pmod{G^\lambda}.$$

Remarque. Si l'on sait manipuler les classes de G/G^λ et les relever dans G, les deux problèmes de la définition II – 7 sont équivalents.

L'introduction de ces nouveaux problèmes permet d'établir la réduction suivante :

Théorème II – 8. *Soient* G *un groupe abélien fini multiplicatif et* k *un entier divisant* $|G|$ *tel que* $|G|$ *et* $|G|/k$ *soient premiers entre eux, on a la réduction polynomiale suivante :*

$$\mathrm{Classe}_{G,k} \overset{\mathscr{P}}{\Longleftarrow} \left(\mathrm{C\text{–}RSA}_{G,k} \wedge \mathrm{Dlog}_{\langle g\rangle}\right).$$

Démonstration. Soit c un élément de G, on note x la réponse d'un oracle résolvant le problème $\text{C–RSA}_{G,k}$. Il existe alors g' dans G^λ tel que

$$c = x^k g'.$$

On en déduit que

$$[\![c]\!]_g = \left[\!\left[c/x^k\right]\!\right]_g = [\![g']\!]_g .$$

Comme g' est un élément de G^λ et que $G^\lambda = \langle g \rangle$, $[\![g']\!]_g$ est en fait le logarithme discret de g' en base g dans $\langle g \rangle$, dont la valeur est retournée par l'oracle correspondant. □

Il est immédiat de vérifier que

$$\text{C–RSA}_{G,k} \overset{\mathscr{P}}{\Longleftarrow} \text{Ordre}_G,$$

puisque connaissant λ, on peut, étant donné c un élément de G, calculer

$$x := c^{k^{-1} \bmod \lambda},$$

qui vérifie bien $x^k \equiv c \pmod{G^\lambda}$. Ceci permet d'établir le corollaire du théorème II – 8 suivant :

Corollaire II – 9. *On a en fait la hiérarchie de problèmes :*

$$\text{Classe}_{G,k} \overset{\mathscr{P}}{\Longleftarrow} \left(\text{C–RSA}_{G,k} \wedge \text{Dlog}_{\langle g \rangle}\right) \overset{\mathscr{P}}{\Longleftarrow} \left(\text{Ordre}_G \wedge \text{Dlog}_{\langle g \rangle}\right).$$

Sécurité sémantique

La définition et le théorème suivants permettent de définir et d'établir les relations entre la sécurité sémantique du système et les problèmes décisionnels liés à la résidualité d'ordre k dans G.

Définition II – 10.

On notera $\text{D–Classe}_{G,k}$ *et on appellera problème décisionnel de classe de résidualité d'ordre* k *dans* G *le problème décisionnel associé à* $\text{Classe}_{G,k}$, i.e., *étant donnés* c *un élément de* G, m *un élément de* $\mathbf{Z}/k\mathbf{Z}$ *et un élément* g *de* G *d'ordre* k, *décider si* $m = [\![c]\!]_g$.

On notera $\text{Rés}_{G,k}$, *et on appellera problème de résidualité d'ordre* k *dans* G, *le problème de reconnaissance des puissances* k*–ièmes de* G *connaissant un élément d'ordre* k *dans* $G^{(1)}$, i.e., *étant donnés* c *un élément de* G *et* g *un élément d'ordre* k *de* G, *décider si* c *est un élément de* G^k.

(1). Cette condition est nécessaire pour établir l'équivalence du théorème II – 11. Dans la pratique, comme on veut que le problème $\text{Dlog}_{\langle g \rangle}$ soit facile, l'écriture des éléments de $\langle g \rangle$ sera simple et il ne sera pas difficile d'exhiber un élément d'ordre k.

Théorème II – 11. *Soient* G *un groupe abélien fini multiplicatif et* k *un entier divisant* $|\mathrm{G}|$ *tel que* $|\mathrm{G}|$ *et* $|\mathrm{G}|/k$ *soient premiers entre eux, on a l'équivalence de problèmes :*

$$\mathrm{D\text{–}Classe}_{\mathrm{G},k} \overset{\mathscr{P}}{\Longleftrightarrow} \mathrm{Rés}_{\mathrm{G},k}.$$

De plus, le système bâti sur $\mathscr{E}_{\mathrm{G},k,g}$ *est sémantiquement sûr si et seulement s'il n'existe pas d'algorithme polynomial pour résoudre le problème de résidualité d'ordre* k *dans* G.

Démonstration. Supposons que l'on possède un oracle pour résoudre $\mathrm{D\text{–}Classe}_{\mathrm{G},k}$. On souhaite résoudre $\mathrm{Rés}_{\mathrm{G},k}$. Étant donnés c dans G et un élément g d'ordre k de G, on tire aléatoirement avec distribution uniforme m dans $\mathbf{Z}/k\mathbf{Z}$ et on soumet à l'oracle le triplet $(g^m c, m, g)$. Vu que

$$\left(c \in \mathrm{G}^k\right) \iff \left(\llbracket g^m c \rrbracket_g = m\right),$$

on retourne la réponse de l'oracle.

Réciproquement, étant donnés un oracle résolvant $\mathrm{Rés}_{\mathrm{G},k}$, un élément c de G, un élément m dans $\mathbf{Z}/k\mathbf{Z}$ et un élément g d'ordre k de G, on souhaite décider si $\llbracket c \rrbracket_g = m$. On soumet la paire $(c\,g^{-m}, g)$ à l'oracle et on retourne la réponse sans changement.

Le résultat concernant la sécurité sémantique est un simple jeu d'écriture. On montre que ces problèmes sont équivalents au problème de reconnaissance de chiffré $\mathrm{REC-CPA}$ (cf. définition I–7). D'une part, décider qu'un élément c de G est le chiffré d'un m dans $\mathbf{Z}/k\mathbf{Z}$ est exactement résoudre $\mathrm{D\text{–}Classe}_{\mathrm{G},k}$. Donc si le système est sémantiquement sûr, le problème $\mathrm{D\text{–}Classe}_{\mathrm{G},k}$ est dur.

D'autre part, résoudre le problème de reconnaissance de chiffré dans le contexte d'une attaque à clairs choisis, c'est être capable d'exhiber un message m, tel que l'on sache discerner des chiffrés aléatoires et des chiffrés de m, *i.e.*, étant donné $c := g^m \rho$, dire si $\rho = c/g^m$ est un élément ou non de G^k. Si on résout le problème de reconnaissance des chiffrés on peut donc résoudre le problème de résidualité d'ordre k. □

Générer des puissances k–ièmes

Première solution : On prend un élément aléatoire de G et on le met à la puissance k. On obtiendra bien une puissance k–ième aléatoire, mais comme le morphisme $f : \mathrm{G} \to \mathrm{G}, x \mapsto x^k$ a pour noyau G^λ , chaque puissance k–ième sera obtenue par k éléments différents.

L'idéal serait de tirer aléatoirement directement dans $\mathrm{G}/\mathrm{G}^\lambda$, puis d'utiliser le fait que le morphisme f passe au quotient en l'isomorphisme

$$\mathrm{G}/\mathrm{G}^\lambda \xrightarrow{\sim} \mathrm{G}^k.$$

Si l'on a un système de représentants des classes modulo G^λ et que l'on sait relever ces classes dans G, on pourra générer aléatoirement un élément de $\mathrm{G}/\mathrm{G}^\lambda$, le relever dans G et enfin le mettre à la puissance k. On obtiendra ainsi une puissance k–ième aléatoire de G. La fonction de chiffrement deviendra alors :

$$\mathscr{E}'_{\mathrm{G},k,g} : \begin{cases} \mathbf{Z}/k\mathbf{Z} \times \mathrm{G}/\mathrm{G}^\lambda & \xrightarrow{\sim} \quad \mathrm{G} \\ (m \quad , \quad \rho) & \longmapsto \quad g^m \rho^k \end{cases}$$

Il est immédiat de vérifier que $\mathscr{E}'_{\mathrm{G},k,g}$ est un isomorphisme de groupes.

Seconde solution : La méthode précédente pour générer des puissances k–ièmes fonctionnera sous réserve que l'on puisse tirer des éléments aléatoires de G ou de G/G^{λ}. Dans le cas contraire, une solution consiste à travailler dans un sous-groupe cyclique de G^k. On publie une puissance k–ième, ρ, ayant pour ordre ℓ, un grand diviseur de λ (le cardinal de G^k). La fonction de chiffrement devient le morphisme injectif suivant :

$$\mathcal{E}''_{G,k,g,\rho} : \begin{cases} \mathbf{Z}/k\mathbf{Z} \times \mathbf{Z}/\ell\mathbf{Z} & \longrightarrow & G \\ (m \quad , \quad r) & \longmapsto & g^m \rho^r \end{cases}$$

Dans la pratique, l'ordre ℓ de ρ ne sera pas public. Pour chiffrer, on disposera d'un majorant de λ ne mettant pas en cause la sécurité du système et les entiers r seront pris inférieurs à ce majorant.

En faisant une analyse similaire à celle faite pour la primitive $\mathcal{E}_{G,k,g}$, on montre que la sécurité sémantique du système bâti sur $\mathcal{E}''_{G,k,g,\rho}$ est équivalente à la difficulté du problème de reconnaissance des éléments de $\langle\rho\rangle$ parmi les éléments de G.

Efficacité du système

L'expansion du système est de

$$\frac{|k|_2 + |\lambda|_2}{|k|_2} = 1 + \frac{|\lambda|_2}{|k|_2},$$

et sera d'autant plus faible que λ est de taille petite devant k.

Pour le chiffrement, comme la sécurité du système ne dépend pas du choix de g, on peut choisir cet élément de manière que le calcul de g^m soit peu coûteux. La production des puissances k–ièmes aléatoires en suivant la première solution requiert essentiellement le calcul d'une exponentiation à la puissance k dans G soit, en moyenne sur les entiers k utilisés, $\frac{3}{2}|k|_2$ multiplications dans G avec l'algorithme classique "square and multiply". Pour la seconde solution, le pire cas prend en moyenne $\frac{3}{2}|\lambda|_2$ multiplications dans G en supposant ℓ de taille proche de celle de λ. On privilégiera donc cette méthode si λ est de taille plus petite que k.

Le coût du déchiffrement est essentiellement celui de l'exponentiation à la puissance λ dans G, celui du calcul du logarithme discret dans $\langle g\rangle$ étant supposé faible. On utilise donc $\frac{3}{2}|\lambda|_2$ multiplications dans G.

Une variante du déchiffrement consiste à utiliser une version effective du théorème II – 8 : si c est le chiffré d'un message m, on a

$$\frac{c}{\left(c^{k^{-1} \bmod \lambda}\right)^k} = g^m.$$

Si le calcul de $c^{k^{-1} \bmod \lambda}$ est fait dans G/G^{λ}, cette méthode de déchiffrement utilise en moyenne $\frac{3}{2}|\lambda|_2$ multiplications dans G/G^{λ} et $\frac{3}{2}|k|_2$ dans G.

Étudions quelle méthode est la plus performante. On estime que le coût d'une opération dans G est le même que celui de

$$\left(\frac{|k|_2 + |\lambda|_2}{|\lambda|_2}\right)^2$$

opérations dans G/G^λ. Avec cette estimation, la seconde méthode de déchiffrement demande

$$\frac{3|k|_2}{2|\lambda|_2^2}\left(|\lambda|_2^2 - |\lambda|_2|k|_2 - |k|_2^2\right)$$

opérations en moins. La seconde méthode sera donc plus performante si et seulement si ce nombre est positif, *i.e.*, si et seulement si

$$|k|_2 < \frac{\sqrt{5}-1}{2}|\lambda|_2 .$$

Au final, la sécurité du système croît avec λ, par contre l'augmentation de la taille de λ accroît le coût du cryptosystème, en particulier celui du déchiffrement. Le rapport de la taille de λ sur celle de k contrôle l'expansion du système.

2. Fonctions trappe probabilistes non homomorphiques

Dans cette section, on introduit deux familles de fonctions trappe probabilistes non homomorphiques. La première peut être vue comme une généralisation avec une perte de structure de celle introduite en sous-section 1.2. La seconde, d'un autre type, est construite à partir de deux fonctions trappe non probabilistes.

2.1. Généralisation de la fonction trappe introduite en 1.2

Idée directrice

Le but de cette généralisation est une amélioration de l'efficacité du chiffrement décrit en 1.2. On a vu que l'étape la plus longue est la création d'une puissance k–ième du groupe G. On a aussi vu (cf. théorème II – 8) que le problème du déchiffrement pouvait se ramener à l'inversion de l'automorphisme :

$$G/\langle g\rangle \xrightarrow{\sim} G/\langle g\rangle, x \mapsto x^k$$

avec les notations de la sous-section 1.2.

L'idée est de remplacer le morphisme $G \to G, x \mapsto x^k$ par une fonction trappe f non nécessairement homomorphique, mais plus rapide à calculer. Le chiffrement sera alors accéléré et le déchiffrement pourra se faire en inversant f dans le groupe quotient $G/\langle g\rangle$. On généralise ainsi la démarche de Catalano *et al.* dans [CGHGN01] qui, à partir du système de Paillier (cf. [Pai99]), créent un système plus rapide. Ces systèmes qui utilisent les

quotients de $\mathbf{Z}$, seront détaillés dans le chapitre IV : respectivement page 84 et page 98. On verra d'autres « instanciations » de cette fonction trappe : un autre système existant, celui de Galindo *et al.* (cf. [GMMV02] et page 101), et un nouveau système particulièrement performant utilisant les quotients de corps quadratiques (cf. sous-section IV–3.3).

Remarque. Si le chiffré d'un message m est de la forme $g^m f(\rho)$ avec ρ un aléa pris dans un ensemble Λ, alors le système obtenu ne sera pas forcément homomorphique. En effet, si l'image de f n'est pas stable, on pourra trouver ρ_1 et ρ_2 tels qu'étant donnés deux messages m_1 et m_2, il n'existe pas de ρ dans Λ tel que

$$g^{m_1} f(\rho_1) g^{m_2} f(\rho_2) = g^{m_1+m_2} f(\rho).$$

Ainsi, $g^{m_1} f(\rho_1) g^{m_2} f(\rho_2)$ n'est pas un chiffré de $m_1 + m_2$ et

$$\mathscr{D}(g^{m_1} f(\rho_1) g^{m_2} f(\rho_2)) \neq m_1 + m_2.$$

On voit maintenant plus en détail comment mettre en œuvre cette idée.

Données publiques

On note G un groupe abélien fini multiplicatif. On se donne un élément g de G d'ordre k. On note H le quotient $\mathrm{G}/\langle g \rangle$ et π la surjection canonique de G dans H. On suppose que le morphisme π est évaluable avec un faible coût. On note Ω un sous-ensemble de G qui sera l'espace des chiffrés et Λ l'espace fini des aléas. Les messages seront des éléments de $\mathbf{Z}/k\mathbf{Z}$.

On se donne une fonction à sens unique f de Λ dans Ω telle que la fonction $\pi \circ f$ soit injective et telle qu'il existe une trappe pour inverser $\pi \circ f$. On suppose de plus que $\langle g \rangle f(\Lambda) = \Omega$. Notons que l'on aura alors $\pi(f(\Lambda)) = \pi(\Omega)$. On aura aussi $k\,|f(\Lambda)| = |\Omega|$. Comme $\pi \circ f$ est injective, f l'est aussi, et on aura en fait $k\,|\Lambda| = |\Omega|$. On donne plus bas une manière de construire la fonction f.

Enfin, on suppose connu un générateur aléatoire d'éléments de Λ ainsi qu'un algorithme de calcul du logarithme discret dans $\langle g \rangle$.

Algorithme de chiffrement

$$\mathscr{E}_{\mathrm{G},f,g} : \begin{cases} \mathbf{Z}/k\mathbf{Z} \times \Lambda \longrightarrow \Omega \\ (m \ , \ \rho) \longmapsto g^m f(\rho) \end{cases}$$

Montrons que la fonction $\mathscr{E}_{\mathrm{G},f,g}$ est bijective. Pour cela, supposons qu'il existe deux couples de $\mathbf{Z}/k\mathbf{Z} \times \Lambda$, $(m_i, \rho_i)_{i\in\{1,2\}}$, tels que

alors,
$$\begin{aligned} g^{m_1} f(\rho_1) &= g^{m_2} f(\rho_2) \qquad (\text{dans G}), \\ \pi(f(\rho_1)) &= \pi(f(\rho_2)) \qquad (\text{dans H}). \end{aligned}$$

On en déduit que $\rho_1 = \rho_2$ car $\pi \circ f$ est injective. Il suit alors que $g^{m_1} = g^{m_2}$ dans G donc $m_1 = m_2$ dans $\mathbf{Z}/k\mathbf{Z}$, car g est d'ordre k. Ainsi, la fonction $\mathscr{E}_{\mathrm{G},f,g}$ est injective et elle est bijective vu l'hypothèse prise sur les cardinaux de Λ et Ω.

Clef privée et déchiffrement

La trappe permettant d'inverser la fonction $\mathscr{E}_{G,f,g}$ est celle permettant d'inverser $\pi \circ f$. Soit c un élément de Ω à déchiffrer, *i.e.*, on doit retrouver m dans $\mathbf{Z}/k\mathbf{Z}$ tel qu'il existe ρ dans Λ tel que $c = g^m f(\rho)$. On commence par calculer $\pi(c)$, puis au moyen de la trappe, on retrouve ρ tel que $(\pi \circ f)(\rho) = \pi(c)$ car $\pi \circ f$ est injective. Par calcul de $c/f(\rho)$, on retrouve g^m et on en déduit m par un calcul de logarithme discret dans $\langle g \rangle$ pour lequel on dispose d'un algorithme public.

Construction de la fonction f

On donne une méthode pratique pour obtenir une fonction f qui satisfasse les hypothèses prises. On choisit G et g tels que l'on connaisse une fonction trappe, $\overline{f}$, permutation d'un ensemble $\overline{\Lambda} \subset H := G/\langle g \rangle$.

On pose ensuite $\Omega := \pi^{-1}(\overline{\Lambda})$ et on définit Λ comme un sous-ensemble de Ω tel que Λ soit un système de représentants des classes de Ω[1]. Ainsi, on a $\pi(\Lambda) = \pi(\Omega) = \overline{\Lambda}$ et π réalise une bijection de Λ sur $\overline{\Lambda}$.

Par construction, on a $k \times |\Lambda| = |\Omega|$. On suppose qu'étant donné un élément $\overline{\rho}$ de $\overline{\Lambda}$, on sait retrouver efficacement le représentant ρ dans Λ tel que $\pi(\rho) = \overline{\rho}$. On résume la situation dans le diagramme :

$$\begin{array}{ccccc}
\Lambda & \hookrightarrow & \Omega & \hookrightarrow & G \\
\downarrow{\scriptstyle \wr} & & \downarrow & & \downarrow{\scriptstyle \pi} \\
\overline{\Lambda} & \xrightarrow[\overline{f}]{\sim} & \overline{\Lambda} & \hookrightarrow & H
\end{array}$$

Par construction, la fonction $\overline{f} \circ \pi$ est une bijection de Λ sur $\overline{\Lambda}$. On note f un relèvement de $\overline{f}$, *i.e.*, une fonction de Λ dans Ω telle que le diagramme suivant commute :

$$\begin{array}{ccc}
\Lambda & \xrightarrow{f} & \Omega \\
\downarrow{\scriptstyle \pi} & & \downarrow{\scriptstyle \pi} \\
\overline{\Lambda} & \xrightarrow[\overline{f}]{} & \overline{\Lambda}
\end{array} \qquad \text{(II.1)}$$

Comme $\pi \circ f = \overline{f} \circ \pi$, la fonction $\pi \circ f$ est bijective et en particulier injective. De plus, la trappe permettant d'inverser $\overline{f}$ permet d'inverser $\pi \circ f$: étant donné $\overline{c}$ un élément de $\overline{\Lambda}$, grâce à la trappe on retrouve $\overline{\rho}$ dans $\overline{\Lambda}$ tel que $\overline{c} = \overline{f}(\overline{\rho})$. On désigne par ρ le représentant de $\overline{\rho}$ dans Λ. On a bien inversé la fonction $\pi \circ f$, étant donné que

$$(\pi \circ f)(\rho) = \left(\overline{f} \circ \pi\right)(\rho) = \overline{f}(\overline{\rho}) = \overline{c}.$$

(1). Ceci n'est pas forcément vrai avec les seules hypothèses du paragraphe « Données publiques », l'ensemble Λ pouvant de plus ne pas être un sous-ensemble de Ω.

Pour finir, Ω étant l'image réciproque de $\overline{\Lambda}$ par π, Ω est stable par multiplication par les éléments de $\langle g \rangle$. On a donc $\langle g \rangle f(\Lambda) \subset \Omega$ et même égalité par cardinalité. Ainsi construite, la fonction f satisfait donc les bonnes hypothèses.

Sécurité

On donne la définition du problème sur lequel est basée la sécurité du schéma.

Définition II – 12. *On notera* $\text{Classe}_{G,f,g}$ *le problème suivant : Étant donné c un élément de Ω, trouver m dans $\mathbf{Z}/k\mathbf{Z}$ tel qu'il existe ρ dans Λ tel que $c = g^m f(\rho)$.*

Pour simplifier l'étude de la difficulté de ce problème, on suppose que la fonction f a été construite à partir de la fonction $\overline{f}$ comme vu précédemment. Avec les définitions et le théorème suivants, on cherche à relier le problème $\text{Classe}_{G,f,g}$ et le problème d'inversion de la fonction $\overline{f}$.

Définition II – 13.

On notera $\text{Hensel}_{G,g}-f$ *le problème suivant : étant donné $\overline{c}$ un élément de $\overline{\Lambda} = \pi(\Omega)$, trouver l'élément c de Ω tel que $c = f(\rho)$ où ρ est l'élément de Λ tel que $\overline{c} = \pi(f(\rho))$.*

On notera $\text{Inv}-\overline{f}$ *le problème d'inversion ponctuelle de la fonction trappe $\overline{f}$,* i.e., *étant donné $\overline{c}$ un élément de $\overline{\Lambda}$, trouver $\overline{\rho}$ dans $\overline{\Lambda}$ tel que $\overline{c} = \overline{f}(\overline{\rho})$.*

Le théorème suivant fait le lien entre ces problèmes.

Théorème II – 14. *Soient* G *un groupe abélien fini multiplicatif, g un élément d'ordre k dans g, $\overline{\Lambda}$ un sous-ensemble du groupe quotient $G/\langle g \rangle$, Λ un sous-ensemble de* G *en bijection avec $\overline{\Lambda}$, et $\overline{f}$ une fonction trappe permutation de $\overline{\Lambda}$. On note π la surjection canonique de* G *sur $G/\langle g \rangle$ et f une fonction de Λ dans $\Omega := \pi^{-1}(\overline{\Lambda})$ telle que $\pi \circ f = \overline{f} \circ \pi$. On a les équivalences et réductions polynomiales suivantes :*

$$\text{Classe}_{G,f,g} \overset{\mathscr{P}}{\iff} \left(\text{Hensel}_{G,g}-f \ \wedge \ \text{Dlog}_{\langle g \rangle}\right) \overset{\mathscr{P}}{\impliedby} \left(\text{Inv}-\overline{f} \ \wedge \ \text{Dlog}_{\langle g \rangle}\right)$$

Démonstration. On montre l'équivalence de gauche. La réduction de droite suivra immédiatement, étant donné que l'on a vu que l'inversion ponctuelle de la fonction $\overline{f}$ permettait d'inverser ponctuellement la fonction $\pi \circ f$ et donc de ramener le problème du déchiffrement à celui du logarithme discret dans $\langle g \rangle$.

Supposons que l'on dispose d'oracles pour résoudre $\text{Hensel}_{G,g}-f$ et $\text{Dlog}_{\langle g \rangle}$. Soit c un élément de Ω. On cherche m, un élément de $\mathbf{Z}/k\mathbf{Z}$, dans la décomposition $c = g^m f(\rho)$ avec ρ dans Λ. On a $\pi(c) = \pi\big(f(\rho)\big)$. On donne $\pi(c)$ à l'oracle résolvant $\text{Hensel}_{G,g}-f$ qui nous renvoie l'élément c' de Ω avec $c' = f(\rho)$. Étant donné c/c', l'oracle résolvant $\text{Dlog}_{\langle g \rangle}$ retourne m.

Réciproquement, on suppose que l'on dispose d'un oracle résolvant $\text{Classe}_{G,f,g}$. Si g' est un élément de $\langle g \rangle$, on tire un élément ρ de Λ au hasard et en envoyant la valeur $g'f(\rho)$ à l'oracle, on récupère m le logarithme de g' en base g. Supposons maintenant que l'on a un élément $\overline{c}$, élément de $\overline{\Lambda}$, pour lequel on veut résoudre le problème $\text{Hensel}_{G,g}-f$.

On tire n au hasard dans $\mathbf{Z}/k\mathbf{Z}$. On note c l'élément de Λ tel que $\pi(c) = \overline{c}$. On envoie à l'oracle l'élément $g^n c$ de Ω (il s'agit bien d'une instance aléatoire pour l'oracle). On obtient alors l'élément m de $\mathbf{Z}/k\mathbf{Z}$ tel que $g^n c = g^m f(\rho)$ avec ρ un élément de Λ. Comme on a $\pi(g^n c) = \overline{c} = \pi\big(f(\rho)\big)$, l'élément $g^{n-m}c$ résout le problème $\mathrm{Hensel}_{G,g} - f$. □

Remarque. L'équivalence de problèmes de ce théorème est aussi valable dans le cadre général du paragraphe « Données publiques » à condition de savoir relever les éléments de $\pi(\Omega)$ dans Ω pour pouvoir faire la preuve de la réciproque. De même, ce théorème est valable pour le cryptosystème de la sous-section 1.2. Avec les notations de cette sous-section, en posant $\Omega := G$, $\overline{\Lambda} := G/G^{\lambda}$ et en prenant pour $\overline{f}$ l'automorphisme $x \mapsto x^k$ de G/G^{λ}, le résultat

$$\mathrm{Classe}_{G,f,g} \overset{\mathscr{P}}{\Longleftarrow} \left(\mathrm{Inv} - \overline{f} \ \wedge \ \mathrm{Dlog}_{\langle g \rangle}\right)$$

est le même que celui obtenu au théorème II – 8, sous réserve que l'on sache manipuler les classes de G/G^{λ} et que l'on sache relever les classes dans G.

Dans la pratique, comme le problème $\mathrm{Dlog}_{\langle g \rangle}$ est considéré facile, la sécurité du système est reliée à celle de la fonction $\overline{f}$. La construction de cette sous-section permet donc de créer une fonction trappe probabiliste à partir d'une fonction trappe déterministe. L'idéal serait de ne pas avoir de perte de sécurité, *i.e.*, que la réduction du théorème II – 14 soit une équivalence de problèmes. En fait, le problème $\mathrm{Hensel}_{G,g} - f$ est une généralisation d'un problème introduit par Catalano, Nguyen et Stern dans [CNS02]. Dans un certain cadre, Catalano *et al.* montrent que le problème d'inversion de la fonction $\overline{f}$ se réduit au problème $\mathrm{Hensel}_{G,g} - f$. On précise ce résultat dans le théorème suivant où l'on note φ l'indicatrice d'Euler.

Théorème II – 15 ("Theorem 1" de [CNS02]). *On se donne un entier RSA n, premier avec $\varphi(n)$, e un entier strictement positif, premier avec $n\varphi(n)$ et ℓ un entier positif. On note*

$$G := \left(\mathbf{Z}/n^{\ell+2}\mathbf{Z}\right)^{\times} \quad et \quad \overline{\Lambda} := H := (\mathbf{Z}/n\mathbf{Z})^{\times} = G/\langle g \rangle,$$

avec $g = 1 + n$. On prend pour $\overline{f}$ l'automorphisme de $H : x \mapsto x^{en^{\ell}}$ et pour f la fonction :

$$\Lambda \longrightarrow \Omega,\ x \longmapsto \left(x^{en^{\ell}} \mod n^{l+2}\right),$$

où $\Lambda := \{k \in \mathbf{N}, 1 \leqslant k \leqslant n, \mathrm{pgcd}(k,n) = 1\}$ et $\Omega := \{k \in \mathbf{N}, 1 \leqslant k \leqslant n^{l+2}, \mathrm{pgcd}(k,n) = 1\}$ sont vus comme des sous-ensembles de G.

On a alors

$$\mathrm{Inv} - \overline{f} \overset{\mathscr{P}}{\Longleftarrow} \mathrm{Hensel}_{G,g} - f.$$

La preuve de ce théorème est intrinsèquement liée au fait que $\overline{f}$ est un morphisme et que l'on peut ramener des équations sur les antécédents et les images de $\overline{f}$ à des équations dans les entiers inférieurs à n. Elle n'a donc pas de généralisation immédiate au cas général.

Le schéma de la preuve est le suivant : soit $\overline{c} \in \overline{\Lambda}$ à inverser. On note $\overline{r}$ l'élément de $\overline{\Lambda}$ à retrouver vérifiant $\overline{c} = \overline{f}(\overline{r})$. On tire au hasard un élément $\overline{a}$ de $\overline{\Lambda}$. On remarque que

$$\overline{f}(\overline{a})\,\overline{c} = \overline{f}(\overline{a}\,\overline{r}).$$

Deux appels à un oracle résolvant $\text{Hensel}_{G,g}-f$ sur les entrées indépendantes $\overline{c}$ et $\overline{f}(\overline{a})\overline{c}$ donnent les valeurs $f(r)$ et $f(\mu)$ de Ω, en notant r et μ les éléments de Λ tels que $\pi(r)=\overline{r}$ et $\pi(\mu)=\overline{a}\,\overline{r}$. On note a l'élément de Λ tels que $\pi(a)=\overline{a}$. À partir des valeurs retournées par l'oracle, du fait de la forme particulière de l'exposant de la fonction f, il est facile de trouver une équation du type

$$ar \equiv \mu(1+zn) \pmod{n^2}$$

où seuls r et μ sont inconnus. Catalano *et al.* montrent alors qu'une réduction d'un réseau de $\mathbf{Z}^2$ suivie d'une recherche exhaustive sur un petit nombre de solutions permettent de retrouver les valeurs de μ et de r dans Λ en temps polynomial. On réfère le lecteur à [CNS02] pour plus de détails.

Sécurité sémantique

L'étude de la sécurité sémantique du système bâti sur la primitive $\mathscr{E}_{G,f,g}$ est similaire à celle faite en sous-section 1.2.

Définition II – 16.

On notera $\text{D-Classe}_{G,f,g}$ *le problème suivant : étant donnés c un élément de* G *et m un élément de $\mathbf{Z}/k\mathbf{Z}$, décider s'il existe un élément ρ de Λ tel que $c = g^m f(\rho)$.*

On notera $\text{Rés}_{G,f,g}$, *le problème de reconnaissance des éléments de $f(\Lambda)$ dans Ω.*

Théorème II – 17. *Soient* G *un groupe abélien fini multiplicatif, g un élément d'ordre k dans g, Λ un ensemble fini et Ω un sous-ensemble de* G. *On note π la surjection canonique de* G *sur* $G/\langle g\rangle$ *et f une fonction à sens unique de Λ dans Ω telle qui $\pi \circ f$ soit une fonction trappe injective. On a l'équivalence de problèmes :*

$$\text{D-Classe}_{G,f,g} \overset{\mathscr{P}}{\iff} \text{Rés}_{G,f,g}\,.$$

De plus, le système bâti sur $\mathscr{E}_{G,f,g}$ est sémantiquement sûr si et seulement s'il n'existe pas d'algorithme polynomial pour résoudre ces problèmes.

La démonstration de ce théorème est identique à celle du théorème II – 11. Cela vient du fait que le système présenté ici et celui de la sous-section 1.2 ont une propriété similaire :

$$\left(c \leftarrow \mathscr{E}_{G,f,g}(m)\right) \iff \left(\frac{c}{g^m} \in f(\Lambda)\right).$$

Efficacité

L'expansion du système est de

$$\frac{||\Omega||_2}{|k|_2} = 1 + \frac{||\Lambda||_2}{|k|_2}$$

et sera d'autant plus faible que le cardinal de l'ensemble des aléas Λ est faible devant k.

Étudions maintenant le coût de ce cryptosystème. Pour le chiffrement, comme dans la sous-section 1.2, on choisira l'élément g afin que le calcul de g^m soit peu coûteux. L'étape la plus longue est alors l'évaluation de $f(\rho)$ dans Ω avec ρ dans Λ.

Le coût du déchiffrement est essentiellement celui de l'inversion de la fonction composée $f \circ \pi$ auquel s'ajoute encore celui de l'évaluation de la fonction f. Concernant la première étape, si la fonction f a été construite à partir d'une fonction trappe $\overline{f}$ permutation d'un ensemble $\overline{\Lambda}$, on a vu que l'inversion de $f \circ \pi$ se ramenait à celle de la fonction $\overline{f}$. Le coût des opérations de chiffrement et déchiffrement sera donc conditionné par l'efficacité de la fonction trappe $\overline{f}$ utilisée.

Pour utiliser la fonction $\mathscr{E}_{G,f,g}$, on a supposé pouvoir construire des éléments aléatoires de Λ. Dans la pratique, on verra que suivant le groupe G utilisé, cette opération peut être délicate à réaliser. Dans certains groupes, il sera même difficile si l'on ne dispose pas des données privées, d'exhiber un élément de Λ. Ce sera le cas, par exemple, si G est le groupe des points d'une courbe elliptique définie sur $\mathbf{Z}/n\mathbf{Z}$ avec n un entier RSA. La solution sera alors de travailler avec un ensemble Λ simple, dans lequel on pourra facilement tirer des éléments au hasard, et de construire les groupes G et H en fonction de chaque élément tiré.

Le système de Galindo *et al.* présenté page 101 et le premier système proposé en sous-section III–3.3 rentrent dans ce cas.

2.2. Une fonction trappe probabiliste bâtie sur deux fonctions trappe déterministes

Dans cette sous-section, on donne une construction d'une fonction trappe probabiliste en combinant deux fonctions trappe déterministes, permutations d'un même groupe. L'idée de cette construction résulte d'un travail en commun avec Damien Vergnaud.

Données publiques

Soit G un groupe abélien fini multiplicatif. On se donne deux fonctions trappe f et h, permutations de G. On suppose connu un générateur aléatoire d'éléments de G. Les messages à chiffrer sont des éléments de G.

Algorithme de chiffrement

On note μ les messages et ρ les aléas. La fonction de chiffrement est :

$$\mathscr{E}_{f,h} : \begin{cases} G \times G \longrightarrow G \times G \\ (\mu\ ,\ \rho) \longmapsto \left(f(\mu\rho)\ ,\ h(\rho^{-1})\right) \end{cases}$$

Les fonctions f et h étant des permutations de G, on voit immédiatement que la fonction $\mathscr{E}_{f,h}$ est une permutation de G^2.

Clef privée et déchiffrement

La trappe permettant d'inverser la fonction $\mathscr{E}_{f,h}$ est le couple de trappes permettant d'inverser f et h. Soit $c := (c_1, c_2)$ un élément de G^2 à déchiffrer. On cherche μ dans G tel qu'il existe ρ dans G tel que $c_1 = f(\mu\rho)$ et $c_2 = h(\rho^{-1})$. On commence par récupérer ρ^{-1} en calculant $h^{-1}(c_2)$. Puis, $f^{-1}(c_1)\rho^{-1}$ donne le message μ.

Sécurité

Définition II – 18. *On notera* $\mathrm{Inv}-f$ *(resp.* $\mathrm{Inv}-h$*) le problème d'inversion ponctuelle de la fonction trappe* f *(resp. de la fonction trappe* h*),* i.e., *étant donné* c *un élément de* G, *trouver* α *dans* G *tel que* $f(\alpha) = c$ *(resp.* $h(\alpha) = c$*).*

Le théorème suivant donne les relations entre la sécurité du système et celle des fonctions f et h.

Théorème II – 19. *Soit* G *un groupe abélien fini multiplicatif et deux fonctions trappe* f *et* h *permutations de* G. *La sécurité du système est équivalente à la difficulté du problème suivant : Étant donné un couple* (c_1, c_2) *de* G^2, *trouver le produit* $\alpha_1\alpha_2$ *dans* G *avec* $f(\alpha_1) = c_1$ *et* $h(\alpha_2) = c_2$. *Ce problème est équivalent au problème d'inversion ponctuelle des fonctions* f *et* h, *noté* $\mathrm{Inv}-f \wedge \mathrm{Inv}-h$.

Démonstration. La première assertion vient du fait que l'application de G^2 dans G^2

$$(\alpha_1, \alpha_2) \longmapsto (\alpha_1\alpha_2, \alpha_2^{-1})$$

est un bijection, d'inverse

$$(\mu, \rho) \longmapsto (\mu\rho, \rho^{-1}).$$

Pour la deuxième assertion, si l'on sait inverser f et h, on sait évidemment résoudre ce problème. Réciproquement, supposons que l'on dispose d'un oracle pour résoudre ce problème. Soit c dans G, on cherche α_1 tel que $f(\alpha_1) = c$. On tire un élément α_2 de G au hasard, et on soumet $\big(c, g(\alpha_2)\big)$ à l'oracle. L'oracle nous donne la valeur $\alpha_1\alpha_2$ de laquelle on déduit α_1. Le problème étant symétrique, on inverse h de la même manière. □

Sécurité sémantique

On remarque que les chiffrés d'un message μ sont les éléments (c_1, c_2) de G^2 tel que $f^{-1}(c_1)h^{-1}(c_2) = \mu$. On donne le problème sur lequel repose la sécurité sémantique du système, en se servant du problème de non-reconnaissance de chiffrés (cf. définition I – 7).

Proposition II – 20. *Le système bâti sur* $\mathscr{E}_{f,h}$ *est sémantique sûr si et seulement s'il est impossible, en temps polynomial, d'exhiber un message* μ *de* G, *pour lequel on saura discerner un élément aléatoire de* G^2 *d'un élément* (c_1, c_2) *tel que* $f^{-1}(c_1)h^{-1}(c_2) = \mu$.

Remarque. Ce problème est trivial si f est un morphisme et $f = h$. En effet, si (c_1, c_2) est un élément de G^2 et μ un élément de G, on aura

$$f^{-1}(c_1)h^{-1}(c_2) = f^{-1}(c_1c_2),$$

et

$$\Big(f^{-1}(c_1)h^{-1}(c_2) = \mu\Big) \iff \Big(c_1c_2 = f(\mu)\Big).$$

Efficacité

L'expansion du système ne dépend pas des données publiques utilisées et sera toujours de 2.

Étudions maintenant le coût de ce cryptosystème. Pour le chiffrement, on a essentiellement besoin d'une évaluation de la fonction f et de la fonction h. Le coût du déchiffrement est essentiellement celui de l'inversion des deux fonctions.

Cette construction très simple permet donc de créer un système probabiliste avec une expansion raisonnable. Suivant les fonctions f et h utilisées, le coût sera aussi relativement faible. La sécurité du système sera parfaitement maîtrisée, seule la sécurité sémantique reposera sur un problème nouveau.

On donnera page 100, un système concret basé sur cette fonction trappe en utilisant le groupe $\mathbf{Z}/n\mathbf{Z}$ où n est un entier RSA. Les fonctions trappe utilisées seront les fonctions RSA et LUC (cf. page 73). On obtiendra ainsi un système probabiliste avec un coût de chiffrement très compétitif.

CHAPITRE III

QUELQUES GROUPES FINIS POUR LA CRYPTOGRAPHIE

Dans ce chapitre, on décrit trois familles de groupes finis qui nous permettront au chapitre IV d'établir des cryptosystèmes en utilisant les fonctions trappe décrites au chapitre II.

On notera, dans tout le chapitre, n un entier positif strictement supérieur à 1. Les groupes que l'on va étudier sont le groupe multiplicatif $(\mathbf{Z}/n\mathbf{Z})^\times$, le groupe des points d'une courbe elliptique définie sur l'anneau $\mathbf{Z}/n\mathbf{Z}$ et enfin le groupe des éléments de norme 1 de l'anneau des entiers d'un corps quadratique modulo n. Dans le chapitre IV, l'entier n utilisé sera systématiquement un carré ou une puissance d'ordre plus élevé d'un entier RSA. Ces cas particuliers seront donc envisagés ici.

Le groupe $(\mathbf{Z}/n\mathbf{Z})^\times$ étant bien connu, on se limite dans la première section à donner quelques résultats sur la complexité des calculs modulo n^{s+1}, avec $s \geqslant 1$, qui nous serviront pour analyser le coût des cryptosystèmes du chapitre IV.

1. Arithmétique modulo n^{s+1}, avec $s \geqslant 1$

Représentation des éléments

Les opérations modulo n^{s+1} peuvent être optimisées en utilisant une représentation des nombres en base n. Si a est un élement de $\mathbf{Z}/n^{s+1}\mathbf{Z}$, on représentera a par le $(s+1)$–uplet d'entiers naturels strictement inférieurs à n, $(a_0, a_1, \ldots, a_{s-1}, a_s)$, tel que

$$a \equiv \sum_{i=0}^{s} a_i n^i \pmod{n^{s+1}}.$$

Notons que a sera inversible si et seulement si a_0 est premier avec n.

Multiplication de deux éléments

Observons tout d'abord la situation quand $s = 1$. Soient a et b deux éléments de $\mathbf{Z}/n^2\mathbf{Z}$, on représente a et b respectivement par (a_0, a_1) et (b_0, b_1), deux couples d'entiers naturels

strictement inférieurs à n. Le produit ab donne

$$ab \equiv a_0 b_0 + (a_1 b_0 + a_0 b_1)n \pmod{n^2}.$$

où $a_1 b_0 + a_0 b_1$ est défini modulo n et $a_0 b_0$ est défini modulo n^2. Pour obtenir la représentation en base n du produit ab, il faut donc exprimer celle de $a_0 b_0$. Comme a_0 et b_0 sont strictement inférieurs à n, la représentation de $a_0 b_0$, notée (c_0, c_1), s'obtient par $c_0 := c \bmod n$ et $c_1 := (c - c_0)/n$, en posant $c := a_0 b_0$ dans $\mathbf{Z}$. Finalement, la représentation du produit ab est

$$\big(c_0, (c_1 + a_1 b_0 + a_0 b_1) \bmod n\big).$$

Elle s'obtient en faisant une multiplication et une division exacte dans $\mathbf{Z}$, une réduction modulaire, deux multiplications et deux additions dans $\mathbf{Z}/n\mathbf{Z}$. La méthode brutale étant estimée à quatre multiplications modulo n.

Dans le cas $s > 1$, on applique un algorithme itératif similaire. On note $(a_0, \ldots, a_s)$ et $(b_0, \ldots, b_s)$ les représentations respectives de a et de b. Supposons que la représentation de ab ait été calculée jusqu'à l'ordre k avec $k \in \{0, s-1\}$. On obtient la représentation à l'ordre $k+1$ en calculant, dans $\mathbf{Z}$,

$$\sum_{i=0}^{k+1} a_i b_{k+1-i} + r,$$

où r est une retenue provenant du calcul des coefficients de rangs inférieurs, similaire à celle que l'on a vue au rang 1. La division euclidienne de cette somme par n donne alors le coefficient à l'ordre $k+1$ et la retenue pour le rang suivant si $k+1 < s$.

Pour le calcul à l'étape k, on estime que l'algorithme coûte autant que $k+1$ multiplications modulo n. Ainsi, une multiplication modulo n^{s+1} coûte autant que

$$\frac{(s+1)(s+2)}{2},$$

multiplications modulo n.

Inversion d'un élément

Étudions l'inversion d'un élément de a de $\big(\mathbf{Z}/n^2\mathbf{Z}\big)^\times$. On note (a_0, a_1) la représentation de a. On trouve la formule suivante :

$$a^{-1} \equiv b_0 - b_0(b_1 + a_1 b_0)n \pmod{n^2}$$

où l'on pose $b_0 := (a_0^{-1} \bmod n)$ et où b_1 est tel que $a_0 b_0 = 1 + b_1 n$ dans $\mathbf{Z}$.

Une inversion modulo n^2 peut donc se faire en une inversion modulo n, une multiplication et une division exacte dans $\mathbf{Z}$, et deux multiplications et une addition dans $\mathbf{Z}/n\mathbf{Z}$. En ne comptant que l'inversion et la multiplication modulaire, on estime le coût de cet algorithme à 12 multiplications modulo n (en estimant que l'inversion coûte autant que 10 multiplications) au lieu de 40 multiplications avec la méthode brutale.

On peut itérer cet algorithme pour calculer un inverse modulo n^{s+1}, pour $s > 1$. Voyons cela. On note $(a_0, \ldots, a_s)$ la représentation de a et $(b_0, \ldots, b_s)$ celle de son inverse. On a vu

que $b_0 := a_0^{-1} \bmod n$. On suppose que la représentation de a^{-1} a été calculée jusqu'à l'ordre k avec $k \in \{0, s-1\}$.

On calcule, dans $\mathbf{Z}$,

$$-b_0\left(r + \sum_{i=1}^{k+1} a_i b_{k+1-i}\right),$$

où r est une retenue provenant du calcul des coefficients de rangs inférieurs. La division euclidienne du résultat de ce calcul par n donne alors le coefficient à l'ordre $k+1$ et la retenue pour le rang suivant si $k+1 < s$.

On estime que le coût de cette étape est équivalent à celui de $k+2$ multiplications modulo n. Le coût total de l'algorithme pour calculer la représentation de a^{-1} modulo n^{s+1} est donc d'une inversion et de $s(s+3)/2$ multiplications modulo n.

2. Courbes elliptiques sur l'anneau $\mathbf{Z}/n\mathbf{Z}$

Le groupe des points d'une courbe elliptique définie sur un corps fini est un objet bien connu en cryptographie. Dans cette section, on étudie les courbes définies sur $\mathbf{Z}/n\mathbf{Z}$, avec n non nécessairement premier. On voit que l'ensemble des points de ces courbes peut toujours être muni d'une structure de groupe, avec une loi explicite.

De tels groupes ont été relativement souvent utilisés, que ce soit pour la factorisation (cf. [Len87b]), la primalité (cf. [AM93]) ou à des fins cryptographiques. Citons [KMOV91], l'adaptation de RSA dans les courbes elliptiques et [Gal02, GMMV02] des variantes du cryptosystème de Paillier que l'on retrouvera dans le chapitre IV.

Cependant, l'étude détaillée de ces groupes n'est jamais vraiment effectuée dans ces références, seuls les éléments nécessaires étant mentionnés. C'est pourquoi on se propose de les étudier plus en détail.

2.1. Définitions

On définit la notion de courbe elliptique sur un anneau en suivant [Len87a]. Soit $\mathscr{R}$, un anneau commutatif unitaire. On dit qu'un ensemble fini d'éléments $(a_i)_{i\in\mathscr{I}}$ de $\mathscr{R}$ est primitif s'il existe un ensemble $(b_i)_{i\in\mathscr{I}}$ d'éléments de $\mathscr{R}$ tel que $\sum_{i\in\mathscr{I}} b_i a_i = 1$. Dans la suite, il sera utile que $\mathscr{R}$ satisfasse la condition (C) suivante :

Pour toute matrice de $\mathscr{M}_{k,\ell}(\mathscr{R})$, avec k et ℓ supérieurs à 2, telle que l'ensemble constitué par ses entrées soit primitif et telle que ses mineurs d'ordre 2 soient nuls, il existe une combinaison $\mathscr{R}$-linéaire de ses lignes constituant un ensemble primitif de ℓ éléments de $\mathscr{R}$.

Notons que cette condition (C) est vraie pour les anneaux finis (cf. [Len87a]).

Définition III – 1. *On définit le plan projectif $\mathbf{P}^2(\mathscr{R})$ comme étant l'ensemble des triplets (X, Y, Z) primitifs de $\mathscr{R}^3$, quotienté par la relation d'équivalence suivante : deux triplets primitifs (X, Y, Z) et (X', Y', Z') sont équivalents si et seulement s'il existe $\lambda \in \mathscr{R}^\times$ tel que $\lambda X = X', \lambda Y = Y', \lambda Z = Z'$. La classe d'équivalence du triplet (X, Y, Z) est notée $(X : Y : Z)$.*

On est alors en mesure de définir le groupe des points d'une courbe elliptique sur un anneau.

Proposition III – 2. *Soit $\mathscr{R}$ un anneau commutatif unitaire tel que $6 \in \mathscr{R}^{\times}$*(1) *et vérifiant la condition* (C)*. Soient a et b deux éléments de $\mathscr{R}$ tels que $\left(4a^3 + 27b^2\right) \in \mathscr{R}^{\times}$. L'ensemble des éléments de $\mathbf{P}^2(\mathscr{R})$ satisfaisant l'équation projective*

$$Y^2Z = X^3 + aXZ^2 + bZ^3, \tag{III.1}$$

noté $E(\mathscr{R}, a, b)$ est un groupe additif commutatif pour la loi d'addition classique définie géométriquement, l'élément neutre étant le point à l'infini $(0:1:0)$.

Pour une preuve, on se réfère à la section 3 de [Len87a]. L'obtention de formules explicites pour la loi de groupe est moins simple que lorsque $\mathscr{R}$ est un corps, les éléments non inversibles de $\mathscr{R}$ rendant la tâche plus ardue.

Dans la suite, on se concentre sur les anneaux nous intéressant. On supposera que n est premier avec 2 et 3 et on notera $E_n(a,b)$ ou plus simplement E_n s'il n'y a pas d'ambiguité, le groupe $E(\mathbf{Z}/n\mathbf{Z}, a, b)$ avec $\operatorname{pgcd}(4a^3 + 27b^2, n) = 1$. Si n est premier, on retrouve la définition classique des courbes elliptiques sur un corps premier.

Si n n'est pas premier, on se sert des restes chinois pour étudier $E_n(a,b)$. En effet, la réduction modulaire conserve les droites des plans projectifs, *i.e.*, envoie une droite passant par deux points donnés sur la droite passant par les deux points réduits. De plus, l'addition dans les courbes elliptiques est définie par des droites. La surjection canonique $\mathbf{Z}/n\mathbf{Z} \to \mathbf{Z}/k\mathbf{Z}$, où $k \mid n$, s'étend donc en un morphisme de groupes de $E_n(a,b) \to E_k(a,b)$. Par conséquent, l'isomorphisme des restes chinois s'étend en un isomorphisme entre groupes associés à des courbes elliptiques. Plus précisément, si n se factorise en produit de nombres premiers p_j distincts :

$$n = \prod_{j \in I} p_j^{\alpha_j},$$

où les α_j, avec j élément de I, sont tous des entiers naturels non nuls, alors on a l'isomorphisme de groupe :

$$E_n(a,b) \xrightarrow{\sim} \prod_{j \in I} E_{p_j^{\alpha_j}}(a,b). \tag{III.2}$$

Pour étudier E_n, il suffit donc d'étudier $E_{p^{s+1}}$ avec s un entier naturel non nul, le cas $s = 0$ étant bien connu. Cette étude se fera en sous-section 2.3. Dans la sous-section suivante, on explicite l'isomorphisme (III.2) dans le cas particulier des groupes E_n où n est un entier RSA.

2.2. Un premier cas particulier : E_n où n est un entier RSA

Ce type de courbe a été assez étudié du fait de son utilisation pour les méthodes de primalité et de factorisation. En cryptographie, Koyama, Maurer *et al.* ont utilisé ces courbes pour adapter le crytosystème RSA dans les courbes elliptiques (cf. [KMOV91]), adaptation que l'on présentera dans le chapitre IV.

On note $n = pq$ et on suppose que a et b sont deux éléments de $\mathbf{Z}/n\mathbf{Z}$ tel que $4a^3 + 27b^2$ soit premier avec n. Le groupe $E_n(a,b)$ défini précédemment est alors le produit des groupes bien connus $E_p(a,b)$ et $E_q(a,b)$.

(1). Cette condition permet de simplifier l'équation de la courbe elliptique et est peu génante pour l'application cryptographique qui nous motive.

Addition dans E_n

Pour additionner des points de E_n, on peut utiliser les formules classiques d'additions dans les courbes elliptiques définies sur des corps (cf. [Sil86], page 58) dans E_p et dans E_q, puis utiliser la loi produit. Pour des utilisations cryptographiques dans lesquelles la factorisation de n est inconnue, cette démarche est impossible.

Dans ce cadre, on utilise en fait un sous-ensemble de E_n, que l'on note E'_n. On exclut certains points de E_n donnant une factorisation de n. Si l'on note $\mathcal{O}_p$ (resp. $\mathcal{O}_q$) le point à l'infini de E_p (resp. de E_q), les points de E_n que l'on exclut sont les points semi-infinis, *i.e.*, les couples de la forme

$$(P_p, \mathcal{O}_q) \quad \text{et} \quad (\mathcal{O}_p, P_q)$$

où P_p et P_q sont des points finis respectifs de E_p et E_q.

En coordonnées affines, on peut aussi voir E'_n comme étant l'ensemble des points (x, y) à coordonnées dans $(\mathbf{Z}/n\mathbf{Z})^2$, satisfaisant dans $\mathbf{Z}/n\mathbf{Z}$ l'équation

$$y^2 = x^3 + ax + b, \tag{III.3}$$

auquel on rajoute un point $\mathcal{O}_n$ à l'infini.

Pour additionner des points de E'_n, on utilise directement dans $\mathbf{Z}/n\mathbf{Z}$ les formules d'additions classiques. En procédant ainsi, on n'obtient pas une loi de groupe, certaines divisions intervenant dans les formules d'additions n'étant pas définies. Par exemple, si on tente de calculer 2P avec $P := (x, y)$ tel que P soit d'ordre 2 dans E_p et d'ordre différent dans E_q. On aura alors $y \equiv 0 \pmod{p}$ et $y \not\equiv 0 \pmod{q}$. La formule du double échouera car y n'est pas inversible.

Cependant la probabilité d'obtenir une opération interdite est négligeable car elle permettrait de factoriser n (c'est d'ailleurs la base de l'algorithme de factorisation présenté dans [Len87b]).

Remarquons que, quand elle est définie, l'addition des points de E'_n coïncide avec celle de E_n.

Ordre et exposant de groupe

D'après l'isomorphisme (III.2), l'ordre de E_n est le produit des ordres de E_p et de E_q. De plus, si on note $\mu := \operatorname{ppcm}\left(|E_p|, |E_q|\right)$, on a

$$\mu.P = \mathcal{O}_n,$$

pour tout point P de E_n.

Dans [KK98], Kunihiro et Koyama ont montré que pour un entier n quelconque, factoriser n est équivalent au calcul de $|E_n|$. Dans le cas particulier qui nous intéresse (n de type RSA), Okamoto et Uchiyama ont montré dans [OU98b] (théorèmes 9 et 10) que les trois problèmes suivants : calcul de $|E_n|$, calcul de μ et factorisation de n sont équivalents si les cardinaux de E_p et E_q vérifient certaines hypothèses (divisibles par de grands nombres premiers…).

Construire des points de E'_n

Dans le cadre d'un système à clef publique dans E'_n, il semble difficile pour le chiffreur, étant donnés a et b, de construire des points de l'ensemble $E'_n(a,b)$ sans connaître la factorisation de n ; s'il choisit un $x \in \mathbf{Z}/n\mathbf{Z}$ et qu'il cherche y satisfaisant (III.3), il doit calculer une racine carrée ce qui est équivalent à factoriser n ; s'il choisit y et qu'il cherche x, il doit trouver une racine d'un polynôme de degré 3 dans $\mathbf{Z}/n\mathbf{Z}$.

Pour élaborer un cryptosystème dans E'_n, on doit donc
— soit rendre public des points de base et faire des exponentiations de ces points pendant le chiffrement ;
— soit construire les courbes en fonction du point à chiffrer.

Cette seconde solution est utilisée dans le cryptosystème KMOV (cf. [KMOV91]) présenté en sous-section IV–1.1.

2.3. Courbes elliptiques modulo p^{s+1}, avec p premier, $p > 3$ et $s \geqslant 1$

Réduction modulo p

Dans [Sil86], Silverman consacre son chapitre VII à l'étude des courbes elliptiques définies sur l'anneau des entiers $\mathscr{R}$ d'un corps local. L'élément fondamental de cette étude est la réduction modulo π où π est une uniformisante, *i.e.*, $\pi\mathscr{R} = \mathscr{M}$ en désignant par $\mathscr{M}$ l'idéal maximal de $\mathscr{R}$.

On va adapter cette étude au cas qui nous intéresse. On considère la courbe $E_{p^{s+1}}(a,b)$, autrement dit la courbe $E(\mathscr{R},a,b)$ avec $\mathscr{R} := \mathbf{Z}/p^{s+1}\mathbf{Z}$, $\mathscr{M} := p\left(\mathbf{Z}/p^{s+1}\mathbf{Z}\right)$ et p ne divisant pas $4a^3 + 27b^2$. On note Π la surjection $\mathbf{Z}/p^{s+1}\mathbf{Z} \to \mathbf{Z}/p\mathbf{Z}$.

Comme vu en sous-section 2.1, le morphisme Π s'étend en un morphisme du groupe $E_{p^{s+1}}$ dans le groupe E_p. Le groupe E_p étant un objet bien connu, l'étude du morphisme Π va nous permettre de mieux appréhender le groupe $E_{p^{s+1}}$. On note $E_1(a,b)$ ou plus simplement E_1, le noyau de Π, *i.e.*, le sous-groupe des points de $E_{p^{s+1}}(a,b)$ au-dessus de l'infini. La propriété suivante est une adaptation de la propriété VII.2.1 de [Sil86].

Proposition III – 3. *Avec les notations précédentes, on a la suite exacte :*

$$0 \longrightarrow E_1 \longrightarrow E_{p^{s+1}} \longrightarrow E_p \longrightarrow 0$$

Démonstration. Le seul élément à montrer est la surjection. Celle-ci découle du lemme de Hensel (cf. [Kob84] "Theorem 3", page 16). Voyons cela. On note

$$f(x,y) := y^2 - x^3 - ax - b,$$

le polynôme associé à l'équation affine de E. Soit $\widetilde{P} := (\tilde{x}_p, \tilde{y}_p)$ un point fini de $E_p(a,b)$ donné en coordonnées affines. Comme cette courbe est non singulière, on a

$$2\tilde{y}_p \not\equiv 0 \pmod{p} \qquad \text{ou} \qquad \left(3\tilde{x}_p^2 + a\right) \not\equiv 0 \pmod{p}.$$

Si on est dans le premier cas, $\tilde{y}_p$ est une racine simple du polynôme $f(\tilde{x}_p, y)$ dans $\mathbf{Z}/p\mathbf{Z}$. Soit $x_p \in \mathbf{Z}/p^{s+1}\mathbf{Z}$ tel que $\tilde{x}_p \equiv x_p \pmod{p}$. Par le lemme de Hensel, il existe un unique $y_p \in \mathbf{Z}/p^{s+1}\mathbf{Z}$ au-dessus de $\tilde{y}_p$ tel que $f(x_p, y_p) = 0$ dans $\mathbf{Z}/p^{s+1}\mathbf{Z}$.

Sinon, vu que la caractéristique est différente de 2, on a $\tilde{y}_p \equiv 0 \pmod{p}$. En choisissant un $y_p \in \mathscr{M}$, on a bien $\tilde{y}_p \equiv y_p \equiv 0 \pmod{p}$. Comme $\left(3\tilde{x}_p^2 + a\right) \not\equiv 0 \pmod{p}$, le lemme de Hensel nous donne un unique $x_p \in \mathbf{Z}/p^{s+1}\mathbf{Z}$ tel que $f(x_p, y_p) \equiv 0 \bmod p^{s+1}$.

Dans les deux cas, on a construit un point $\mathrm{P} := (x_p, y_p)$ de $\mathrm{E}_{p^{s+1}}$ au-dessus de $\widetilde{\mathrm{P}}$. Le point à l'infini de E_p peut être relevé par celui de $\mathrm{E}_{p^{s+1}}$. Ainsi, le morphisme Π est bien surjectif. □

Le groupe $\mathrm{E}_p(a, b)$ étant bien connu, on va s'intéresser au sous-groupe $\mathrm{E}_1(a, b)$. Dans le cas $s = 1$, on peut facilement recenser les éléments de $\mathrm{E}_1(a, b)$. Comme ces éléments se réduisent modulo p au point à l'infini $(0 : 1 : 0)$, ils sont *a priori* de la forme $(kp : 1 : k'p)$, k et k' étant des éléments de $\mathbf{Z}/p\mathbf{Z}$. Comme ces points doivent de plus vérifier l'équation projective (III.1), on trouve que k' est nécessairement nul et k quelconque.

Ainsi, $\mathrm{E}_1(a, b) = \{(kp : 1 : 0), k \in \mathbf{Z}/p\mathbf{Z}\}$, *i.e.*, $\mathrm{E}_1(a, b) = \{(z : 1 : 0), z \in \mathscr{M}\}$. Remarquons que les éléments de E_1 ne dépendent ni de a, ni de b. De plus, on a $|\mathrm{E}_1| = p$ et

$$\left|\mathrm{E}_{p^2}(a, b)\right| = p \times \left|\mathrm{E}_p(a, b)\right|.$$

Dans la suite, on va voir que dans le cas s quelconque, E_1 est isomorphe au groupe formel de la courbe elliptique $\mathrm{E}_{p^{s+1}}$. Ceci nous fournira à la fois à une description de ses éléments et de sa loi de groupe. On établira aussi son cardinal, ce qui nous donnera celui de $\mathrm{E}_{p^{s+1}}$, en généralisant la formule trouvée dans le cas $s = 1$.

Définissons tout d'abord le groupe formel d'une courbe elliptique.

Groupe formel

On suit maintenant le chapitre IV de [Sil86]. Commençons par donner la définition d'un groupe formel.

Définition III – 4. *Un groupe formel $\mathscr{F}$ commutatif à un paramètre défini sur un anneau $\mathscr{R}$ est une série formelle* $\mathrm{F}(\mathrm{X}, \mathrm{Y}) \in \mathscr{R}[\![\mathrm{X}, \mathrm{Y}]\!]$ *telle que :*

- $\mathrm{F}(\mathrm{X}, \mathrm{Y}) = \mathrm{X} + \mathrm{Y} +$ *des termes de degrés supérieurs à 2 ;*
- $\mathrm{F}(\mathrm{X}, 0) = \mathrm{X}$ *(élément neutre) ;*
- *il existe une unique série formelle* $i(\mathrm{T}) \in \mathscr{R}[\![\mathrm{T}]\!]$, *telle que* $\mathrm{F}\big(\mathrm{T}, i(\mathrm{T})\big) = 0$ *(opposé) ;*
- $\mathrm{F}\big(\mathrm{X}, \mathrm{F}(\mathrm{Y}, \mathrm{Z})\big) = \mathrm{F}\big(\mathrm{F}(\mathrm{X}, \mathrm{Y}), \mathrm{Z}\big)$ *(associativité) ;*
- $\mathrm{F}(\mathrm{X}, \mathrm{Y}) = \mathrm{F}(\mathrm{Y}, \mathrm{X})$ *(commutativité).*

La série $\mathrm{F}(\mathrm{X}, \mathrm{Y})$ *est appelée loi de groupe formel de $\mathscr{F}$.*

Un exemple immédiat de groupe formel est le groupe formel additif : $\mathrm{F}(\mathrm{X}, \mathrm{Y}) = \mathrm{X} + \mathrm{Y}$. Une courbe elliptique permet également de définir une loi formelle. Soit $\mathrm{E}(a, b)$ une courbe elliptique définie sur un corps quelconque de caractéristique différente de 2 et 3. Étant donné un point $(\mathrm{X}, \mathrm{Y}, \mathrm{Z})$ de E d'ordre différent de 2, on se place dans le plan $(\mathrm{Y} = 1)$ en

divisant[1], par Y (comme le point est d'ordre différent de 2, Y est bien inversible). Les points de E d'ordre différent de 2 peuvent donc être écrit sous la forme (X : 1 : Z). On les note (z, w) de telle sorte que $z = X/Y$ et $w = Z/Y$. L'opposé de (z, w) est alors $(-z, -w)$, le point à l'infini s'écrit $(0, 0)$. L'équation de E devient

$$w = z^3 + azw^2 + bw^3. \tag{III.4}$$

On réinjecte cette expression de w dans le membre de droite :

$$w = z^3 + az\left(z^3 + azw^2 + bw^3\right)^2 + b\left(z^3 + azw^2 + bw^3\right)^3.$$

En réitérant le processus, la méthode converge (cf. Proposition IV.1.1 de [Sil86]), on trouve une expression de w sous forme d'une série formelle $w(z) \in \mathbf{Z}[a, b][\![z]\!]$:

$$w(z) = z^3 + az^7 + bz^9 + 2a^2z^{11} + 5abz^{13} + (5a^3 + 3b^2)z^{15} + 21a^2bz^{17} + \dots \tag{III.5}$$

De plus, cette expression satisfait $w(z) = z^3 + azw(z)^2 + bw(z)^3$, à rapprocher de l'équation (III.4). Ainsi, les couples $\big(z, w(z)\big)$ sont des points « formels » de la courbe elliptique dans le plan $(Y = 1)$.

On note maintenant w pour $w(z)$. Deux points (z_1, w_1) et (z_2, w_2) peuvent être additionnés dans le plan $(Y = 1)$ avec la technique géométrique habituelle. Comme on connait l'expression de w en fonction de z, on a juste à calculer l'expression de l'abscisse du résultat. On considère la droite Δ passant par les points (z_1, w_1) et (z_2, w_2). Sa pente est

$$\lambda := \frac{w_2 - w_1}{z_2 - z_1},$$

ce qui nous donne une série formelle en z_1, z_2 en remplaçant w_1 et w_2 par leurs expressions. On en déduit l'équation de Δ et on cherche les coordonnées d'un troisième point d'intersection (z_3, w_3). On trouve z_3 sous forme d'une série formelle en z_1, z_2. L'ordonnée w_3 est obtenue par le calcul de $w(z_3)$. La somme formelle de deux points est alors l'opposé de (z_3, w_3) soit $(-z_3, -w_3)$ et $-w_3 = w(-z_3)$ car w est impaire. L'abscisse de l'addition de (z_1, w_1) et (z_2, w_2) est donc donnée par une série formelle F en (z_1, z_2) dont voici l'expression :

$$\begin{aligned} \mathrm{F}(z_1, z_2) = z_1 + z_2 &- a\left(2z_1z_2^4 + 4z_1^2z_2^3 + 4z_1^3z_2^2 + 2z_1^4z_2\right) \\ &- b\left(3z_1z_2^6 + 9z_1^2z_2^5 + 15z_1^3z_2^4 + 15z_1^4z_2^3 + 9z_1^5z_2^2 + 3z_1^6z_2\right) \\ &+ a^2\left(-2z_1z_2^8 + 8z_1^3z_2^6 + 16z_1^4z_2^5 + 16z_1^5z_2^4 + 8z_1^6z_2^3 - 2z_1^8z_2\right) \\ &+ \cdots \end{aligned} \tag{III.6}$$

L'abscisse de l'opposé d'un point (z_1, w_1) est donnée par :

$$i(z_1) := -z_1.$$

(1). Silverman divise par $-Y$ ce qui est plus commode dans son cas : il considère des courbes définies sur des corps de caractéristique quelconque avec, par conséquent, une équation plus complexe.

Les séries F et i permettent ainsi de définir un groupe formel associé à la courbe E, on le note $\widehat{\mathrm{E}}$.

À partir des points formels (z, w), on est tenté de produire de véritables points de la courbe elliptique en donnant à z des valeurs du corps de base. Plus généralement, étant donnés un groupe formel muni d'une loi $\mathrm{F}(\mathrm{X},\mathrm{Y}) \in \mathscr{R}[\![\mathrm{X},\mathrm{Y}]\!]$ et un ensemble $\mathscr{M} \subset \mathscr{R}$, on souhaiterait établir une structure de groupe sur $\mathscr{M}$ à l'aide de la loi formelle F. Pour que ce procédé fonctionne, il faut que les séries convergent. C'est le cas si $\mathscr{R}$ est un anneau local complet et si $\mathscr{M}$ est son idéal maximal. On obtient ainsi un groupe associé au groupe formel (cf. section IV.3 de [Sil86]).

Dans notre cas, comme $\mathscr{R} = \mathbf{Z}/p^{s+1}\mathbf{Z}$ et $\mathscr{M} = p(\mathbf{Z}/p^{s+1}\mathbf{Z})$, le procédé va aussi fonctionner (les séries vont évidemment converger). On note $\widehat{\mathrm{E}}(\mathscr{M})$ l'ensemble $\mathscr{M}$ muni de la loi additive suivante :

$$\begin{array}{rcll} z_1 \oplus z_2 & := & \mathrm{F}(z_1, z_2) & \text{pour } z_1 \text{ et }, z_2 \text{ dans } \mathscr{M}, \\ \ominus z_1 & := & i(z_1) = -z_1 & \text{pour } z_1 \text{ dans } \mathscr{M}. \end{array}$$

Cette loi confère à $\widehat{\mathrm{E}}(\mathscr{M})$ une structure de groupe commutatif.

On relie maintenant le groupe $\widehat{\mathrm{E}}(\mathscr{M})$ muni de la loi héritée du groupe formel au sous-groupe E_1 de $\mathrm{E}_{p^{s+1}}$.

Conséquences sur E_1 et $\mathrm{E}_{p^{s+1}}$

Si z est un élément de $\mathscr{M}$, la série $w(z)$ va converger dans $\mathbf{Z}/p^{s+1}\mathbf{Z}$. Ainsi, on peut produire un point $(z, w(z))$ satisfaisant (III.4). En coordonnées projectives, ce point s'écrit $(z : 1 : w(z))$. Vu l'expression de $w(z)$, ce point est un élément de E_1, le sous-groupe de $\mathrm{E}_{p^{s+1}}$ constitué par les éléments qui se réduise au point à l'infini modulo p.

On a en fait un morphisme de groupe noté $\psi : \widehat{\mathrm{E}}(\mathscr{M}) \to \mathrm{E}_1$, car la loi F a été définie comme celle de E. La propriété suivante, qui est une adaptation de la proposition VII.2.2 de [Sil86], montre que ψ est un isomorphisme.

Proposition III – 5. *Avec les notations du paragraphe précédent, le morphisme de groupe*

$$\begin{array}{rccc} \psi : & \widehat{\mathrm{E}}(\mathscr{M}) & \longrightarrow & \mathrm{E}_1 \\ & z & \longmapsto & (z : 1 : w(z)) \end{array}$$

est un isomorphime.

Démonstration. Par ce qui précède le morphisme ψ est bien défini. L'injectivité de ψ est immédiate. Pour montrer que l'on a bien un isomorphisme, on considère l'application :

$$\begin{array}{rccc} \phi : & \mathrm{E}_1 & \longrightarrow & \widehat{\mathrm{E}}(\mathscr{M}) \\ & (z : 1 : z') & \longmapsto & z \end{array}$$

où z et z' sont deux éléments de $\mathscr{M}$. Cette application est bien définie et c'est un morphisme toujours par définition de la loi de groupe de $\widehat{\mathrm{E}}(\mathscr{M})$. L'élément neutre de $\widehat{\mathrm{E}}(\mathscr{M})$

étant 0, le noyau de ce morphisme est $(0 : 1 : z')$, où $z' \in \mathscr{M}$. En reportant dans l'équation (III.4), on trouve $z' = b(z')^3$ dans $\mathbf{Z}/p^{s+1}\mathbf{Z}$. En factorisant, on obtient

$$z'(1 - b(z')^2) = 0,$$

toujours dans $\mathbf{Z}/p^{s+1}\mathbf{Z}$. Comme z' est un élément de $p\left(\mathbf{Z}/p^{s+1}\mathbf{Z}\right)$, l'élément $1 - bz'^2$ est inversible. On en déduit que $z' = 0$. L'application ϕ est donc injective. Comme $\psi \circ \phi$ est l'identité de $\widehat{\mathrm{E}}(\mathscr{M})$, on obtient l'isomorphisme. □

Conséquences : Comme $|\mathrm{E}_1| = |\mathscr{M}| = p^s$, cette proposition nous permet de calculer l'ordre de $\mathrm{E}_{p^{s+1}}$:

$$\left|\mathrm{E}_{p^{s+1}}\right| = p^s \times \left|\mathrm{E}_p\right|. \tag{III.7}$$

On en déduit également l'expression des éléments de E_1 :

$$\mathrm{E}_1 = \left\{ (z : 1 : w(z)),\, z \in p\left(\mathbf{Z}/p^{s+1}\mathbf{Z}\right) \right\},$$

ainsi qu'une formule explicite pour l'addition de ses éléments :

$$(z_1 : 1 : w(z_1)) + (z_2 : 1 : w(z_2)) = (z_1 \oplus z_2 : 1 : w(z_1 \oplus z_2)),$$

où z_1 et z_2 sont des éléments de $\mathscr{M}$.

Exemple pour $s = 1$: On a alors $\mathscr{M} = p\left(\mathbf{Z}/p^2\mathbf{Z}\right) = \left\{kp,\, k \in \mathbf{Z}/p\mathbf{Z}\right\}$. Comme pour tout $z \in \mathscr{M}$, $w(z) \equiv 0 \pmod{p^2}$, on retrouve l'expression des éléments de E_1, *i.e.*, $\mathrm{E}_1 = \left\{(kp : 1 : 0),\, k \in \mathbf{Z}/p\mathbf{Z}\right\}$.

Voyons comment la loi du groupe formel permet d'obtenir l'addition explicite des éléments de E_1. Si z_1 et z_2 sont deux éléments de $\mathscr{M}$, l'expression (III.6) donne

$$\mathrm{F}(z_1, z_2) \equiv z_1 + z_2 \pmod{p^2}.$$

On écrit z_1 et z_2 sous la forme $z_i = k_i p$ avec $k_i \in \mathbf{Z}/p\mathbf{Z}$, pour $i \in \{1, 2\}$. On a alors

$$\mathrm{F}(z_1, z_2) \equiv (k_1 + k_2)\, p \pmod{p^2},$$

le calcul $k_1 + k_2$ étant défini modulo p. Ainsi, on obtient un isomorphisme effectif

$$\mathbf{Z}/p\mathbf{Z} \;\cong\; \widehat{\mathrm{E}}(\mathrm{M}).$$

Cet isomorphisme se prolonge à E_1. Le sous-groupe E_1 est donc cyclique d'ordre p et si on note $\mathscr{O}_k := (kp : 1 : 0)$, avec $k \in \mathbf{Z}/p\mathbf{Z}$, ses éléments, sa loi de groupe est donnée par :

$$\mathscr{O}_{k_1} + \mathscr{O}_{k_2} = \mathscr{O}_{k_1 + k_2},$$

pour k_1 et k_2 dans $\mathbf{Z}/p\mathbf{Z}$ et

$$-\mathscr{O}_{k_1} = \mathscr{O}_{-k_1},$$

pour k_1 dans $\mathbf{Z}/p\mathbf{Z}$.

2.4. Addition dans $E_{p^{s+1}}$, avec p premier, $p > 3$ et $s \geqslant 1$

Dans cette sous-section, on donne les formules explicites d'addition dans le groupe $E_{p^{s+1}}$. Dans un premier temps, on donne des formules en coordonnées affines dérivées des formules d'additions usuelles dans le groupe des points d'une courbe elliptique définie sur un corps. Ensuite, on donne un système complet en coordonnées projectives. Ce système va nous permettre de donner des formules pour les cas non traités en coordonnées affines. Pour finir, à titre d'exemple, on donne un algorithme complet d'addition en coordonnées affines dans E_{p^2}.

Dans cette sous-section, on considère le groupe $E_{p^{s+1}}(a, b)$, où a et b sont deux éléments de $\mathbf{Z}/p^{s+1}\mathbf{Z}$ tels que $(4a^3 + 27b^2)$ soit inversible.

Formules en coordonnées affines (1)

En coordonnées affines, un point fini P de $E_{p^{s+1}}$ est noté (x, y), où x et y sont des éléments de $\mathbf{Z}/p^{s+1}\mathbf{Z}$ tels que

$$f(x, y) := y^2 - (x^3 + ax + b) = 0,$$

dans $\mathbf{Z}/p^{s+1}\mathbf{Z}$. Les points restants de $E_{p^{s+1}}(a, b)$ sont les points au-dessus de l'infini, constituant le sous-groupe E_1 de $E_{p^{s+1}}$. Par la propriété III – 5, les éléments de E_1 sont, en coordonnées projectives, de la forme $(z : 1 : w(z))$, avec $z \in p(\mathbf{Z}/p^{s+1}\mathbf{Z})$. On note $\mathcal{O}_k$, avec $k \in \mathbf{Z}/p^s\mathbf{Z}$ le point $(kp : 1 : w(kp))$. Dans la suite, étant donné un point P de $E_{p^{s+1}}$, on note $\widetilde{P}$, le point de E_p correspondant à P réduit modulo p.

Addition de points finis Soient $P_1 := (x_1, y_1)$ et $P_2 := (x_2, y_2)$ deux points finis de $E_{p^{s+1}}$ à additionner. La droite passant par P_1 et P_2 a pour pente

$$\lambda := \frac{y_2 - y_1}{x_2 - x_1}. \tag{III.8}$$

Cette formule ne fonctionne pas si $x_1 \equiv x_2 \pmod{p}$: cas où les points réduits modulo p, $\widetilde{P}_1$ et $\widetilde{P}_2$, sont égaux ou opposés dans E_p. Si la formule fonctionne, l'opposé du troisième point d'intersection (qui est la somme de P_1 et P_2) a alors pour coordonnées :

$$\begin{aligned} x_3 &:= \lambda^2 - x_1 - x_2, \\ y_3 &:= \lambda(x_1 - x_3) - y_1. \end{aligned} \tag{III.9}$$

On peut trouver une formule pour λ qui marche dans d'autres cas. On multiplie le dénominateur et le numérateur de la quantité λ trouvée précédemment par $y_2 + y_1$. Ceci donne

$$\lambda = \frac{y_2^2 - y_1^2}{(x_2 - x_1)(y_2 + y_1)} = \frac{(x_2 - x_1)(x_2^2 + x_2x_1 + x_1^2 + a)}{(x_2 - x_1)(y_2 + y_1)} = \frac{x_2^2 + x_2x_1 + x_1^2 + a}{y_2 + y_1}. \tag{III.10}$$

Remarquons que si $P_1 = P_2$, on retrouve l'expression de la pente de la tangente en P_1. Cette seconde formule donnant λ échoue si $y_2 \equiv -y_1 \pmod{p}$. Ainsi, la première expression de λ permet d'additionner des points finis P_1 et P_2 au-dessus de deux points $\widetilde{P}_1$ et $\widetilde{P}_2$ distincts et non opposés dans E_p. La seconde permet, avec un surcoût calculatoire, de traiter en plus le cas de l'addition de points finis P_1 et P_2 au-dessus de deux points $\widetilde{P}_1$ et $\widetilde{P}_2$ égaux, mais d'ordre différent de 2 dans E_p (afin que $\tilde{y}_2 + \tilde{y}_1$ soit non nul dans $\mathbf{Z}/p\mathbf{Z}$).

Pour couvrir tous les cas d'additions de points finis, il reste à trouver des formules pour le cas où $\widetilde{P}_1$ et $\widetilde{P}_2$ sont opposés dans E_p (cela comprend le cas où ils sont égaux et d'ordre 2). Dans ce cas, on doit trouver un point au dessus de l'infini. On donnera page 47 une formule pour ce cas.

Addition de points au-dessus de l'infini Soient $\mathcal{O}_{k_1}$ et $\mathcal{O}_{k_2}$ deux points de E_1. Si $\mathcal{O}_{k_3}$ désigne la somme de ces deux points, l'élément k_3 de $\mathbf{Z}/p^s\mathbf{Z}$ est donné par la loi du groupe formel. On note $z_3 := k_3 p$ dans $\mathbf{Z}/p^{s+1}\mathbf{Z}$. On a alors

$$z_3 = z_1 \oplus z_2 = \mathrm{F}(z_1, z_2) = z_1 + z_2 + a(-2z_1z_2^4 - 4z_1^2z_2^3 - 2z_1^4z_2) + \dots$$

avec $z_1 := k_1 p$ et $z_2 := k_2 p$.

Addition d'un point fini et d'un point au-dessus de l'infini On traitera ce cas en toute généralité en page 47. Dans la section 2 de [Gal02], Galbraith donne une formule qui va fonctionner dans ce cas si le point fini n'est pas au-dessus d'un point d'ordre 2, *i.e.*, si son ordonnée est inversible. En fait, cette formule fonctionne dans un cadre plus général. Voyons cela plus en détail. On se ramène aux points du plan projectif de la forme (X : 1 : Z) en effectuant le même changement de variable que pour le définition du groupe formel d'une courbe elliptique : on note (z, w) le nouveau système de coordonnées ; si $P := (x, y)$ est un point fini en coordonnées affines classiques, on effectue le changement de variables $z = x/y$ et $w = 1/y$. Il faut donc que ce point soit au-dessus d'un point d'ordre différent de 2. Un point $\mathcal{O}_k$ de E_1 a pour coordonnées dans ce système le couple $\big(kp, w(kp)\big)$, la série w étant donnée par l'expression (III.5).

L'addition de deux points (z_1, w_1) et (z_2, w_2) se fait toujours par l'interprétation géométrique. Si on note (z_3, w_3) la somme des deux points, on a

$$\begin{aligned} z_3 &= z_1 + z_2 + (w_1 + \lambda z_1)(2a\lambda + 3b\lambda^2)/(1 + a\lambda^2 + b\lambda^3), \\ w_3 &= \lambda(z_3 + z_1) - w_1, \end{aligned} \tag{III.11}$$

où $\lambda = (w_1 - w_2)/(z_1 - z_2)$.

Remarquons qu'avec cette formule, on ne peut pas additionner deux points au-dessus de l'infini : la division dans l'expression de λ ne serait alors pas définie. Ces formules requièrent quatre multiplications, un carré et deux inversions modulaires, ce qui les rend beaucoup plus lentes que celles présentées précédement. Il est donc préférable de ne les utiliser que pour les cas d'additions non couverts, à savoir additionner un point fini (avec y inversible) et un point au-dessus de l'infini.

Pour un point fini, la conversion du résultat en coordonnées (x, y) classiques se fait par $(x, y) = (z/w, 1/w)$, en utilisant une inversion et une multiplication. Galbraith donne aussi dans [Gal02] une formule pour le double, qui ne permet pas de traiter de nouveaux cas.

Au final, les cas qui n'ont pas été couverts en coordonnées affines sont l'addition de deux points finis au-dessus de deux points opposés, et l'addition d'un point fini au-dessus d'un point d'ordre 2 avec un point infini. On reviendra sur ces deux cas manquants après avoir établit un système complet en coordonnées projectives.

Formules en coordonnées projectives

Pour obtenir des formules en coordonnées projectives, le plus simple est de partir des formules affines, d'effectuer les changements de variables $x = \mathrm{X}/\mathrm{Z}$ et $y = \mathrm{Y}/\mathrm{Z}$, puis de multiplier par la bonne quantité afin de ne plus avoir de dénominateur. Par rapport aux formules affines, on gagne la possibilité d'obtenir l'infini comme résultat et on n'effectue aucune inversion dans $\mathbf{Z}/p^{s+1}\mathbf{Z}$.

Cependant, en procédant de cette manière, même dans le cas d'une courbe elliptique sur un corps, on ne peut pas additionner des points finis avec des points à l'infini car la formule affine est obtenue grâce à une droite passant par deux points finis. Une meilleure solution est d'établir des formules d'additions plus générales en faisant un peu de géométrie projective.

Faisons un peu de géométrie projective Soit K un corps, on se place dans le plan projectif $\mathbf{P}^2(\mathrm{K})$. On cherche les intersections d'une cubique, *i.e.*, l'ensemble des points de $\mathbf{P}^2(\mathrm{K})$ solutions d'une équation homogène de degré 3, et d'une droite. On a le résultat suivant (cité dans [JQ01], afin d'obtenir des formules d'additions performantes dans le cas particulier des courbes elliptiques dites Hessiennes ; on doit apparemment ce théorème à Desboves [Des86]).

Théorème III – 6. *Soient* E *une cubique sur le corps* K, F *le polynôme homogène de degré 3 la définissant,* $\mathrm{P}_1 := (\mathrm{X}_1 : \mathrm{Y}_1 : \mathrm{Z}_1)$ *et* $\mathrm{P}_2 := (\mathrm{X}_2 : \mathrm{Y}_2 : \mathrm{Z}_2)$ *deux points distincts de* E *donnés en coordonnées projectives. Les coordonnées projectives du troisième point d'intersection de la droite* $(\mathrm{P}_1\mathrm{P}_2)$ *et de* E *sont*

$$(\lambda \mathrm{X}_1 + \mu \mathrm{X}_2 : \lambda \mathrm{Y}_1 + \mu \mathrm{Y}_2 : \lambda \mathrm{Z}_1 + \mu \mathrm{Z}_2),$$

avec $\lambda = \langle \partial \mathrm{F}(\mathrm{P}_2), \mathrm{P}_1 \rangle$ *et* $\mu = -\langle \partial \mathrm{F}(\mathrm{P}_1), \mathrm{P}_2 \rangle$, *en notant* ∂F *le gradient de* F.

Démonstration. La droite passant par P_1 et P_2 a pour équation

$$\begin{cases} \mathrm{X} &= \lambda \mathrm{X}_1 + \mu \mathrm{X}_2 \\ \mathrm{Y} &= \lambda \mathrm{Y}_1 + \mu \mathrm{Y}_2 \\ \mathrm{Z} &= \lambda \mathrm{Z}_1 + \mu \mathrm{Z}_2 \end{cases}$$

paramétrée par λ et μ, deux éléments de K. En fait, comme on est dans le plan projectif, seul le rapport λ/μ compte dans cette paramétrisation. Le point P_1 correspond à un rapport

infini et le point P_2 à un rapport nul. Pour trouver le troisième point d'intersection de la droite (P_1P_2) avec E, on reporte ces relations dans F. On voit facilement que

$$\begin{aligned} F(\lambda P_1 + \mu P_2) = \mu^3 F(P_2) + \mu^2\lambda \left(X_1 \frac{\partial F}{\partial X}(P_2) + Y_1 \frac{\partial F}{\partial Y}(P_2) + Z_1 \frac{\partial F}{\partial Z}(P_2) \right) \\ + \mu\lambda^2 \left(X_2 \frac{\partial F}{\partial X}(P_1) + Y_2 \frac{\partial F}{\partial Y}(P_1) + Z_2 \frac{\partial F}{\partial Z}(P_1) \right) + \lambda^3 F(P_1). \end{aligned}$$

Comme P_1 et P_2 sont deux points de E, outre les solutions nulles, on obtient

$$\lambda \langle \partial F(P_1), P_2 \rangle + \mu \langle \partial F(P_2), P_1 \rangle = 0.$$

Le couple $\left(\langle \partial F(P_2), P_1 \rangle, - \langle \partial F(P_1), P_2 \rangle \right)$ est donc un couple solution, ce qui prouve le théorème. □

Addition au-dessus de points distincts modulo p Le théorème précédent s'étend à notre situation et permet d'établir une formule d'addition de deux points de $E_{p^{s+1}}$ au-dessus de deux points distincts de E_p (finis ou infinis).

Soient $P_1 := (X_1 : Y_1 : Z_1)$ et $P_2 := (X_2 : Y_2 : Z_2)$ deux points de $E_{p^{s+1}}$ tels que $\widetilde{P}_1 \neq \widetilde{P}_2$ dans $\mathbf{P}^2(\mathbf{Z}/p\mathbf{Z})$. Les coordonnées $(X_3 : Y_3 : Z_3)$ du point P_3 somme de P_1 et P_2 sont données par :

$$\begin{array}{rcl} X_3 & = & \lambda X_1 + \mu X_2 \\ Y_3 & = & -\lambda Y_1 - \mu Y_2 \\ Z_3 & = & \lambda Z_1 + \mu Z_2 \end{array} \qquad \text{(III.12)}$$

où $\lambda := -X_1(3X_2^2 + aZ_2^2) + (2Y_1Y_2)Z_2 + Z_1(Y_2^2 - 2aX_2Z_2 - 3bZ_2^2)$ et $\mu := X_2(3X_1^2 + aZ_1^2) - (2Y_1Y_2)Z_1 - Z_2(Y_1^2 - 2aX_1Z_1 - 3bZ_1^2)$.

La complexité de cette addition est de 15 multiplications et 6 carrés dans $\mathbf{Z}/p^{s+1}\mathbf{Z}$. Ceci est à comparer aux 12 multiplications et 2 carrés de la formule classique d'addition en coordonnées projectives déduite de l'addition en affine.

Il nous reste à traiter le cas d'addition de deux points de $E_{p^{s+1}}$ au-dessus de deux points égaux dans E_p.

Addition au-dessus de points finis, non forcément distincts modulo p On adapte la formule (III.9) avec l'expression (III.10) de λ en coordonnées projectives. En utilisant la méthode brutale (en posant $x := X/Z$ et $y := Y/Z$) on aboutit à des formules avec 17 multiplications et 3 carrés. Dans la section 3 de [BJ02], en utilisant la commutativité de l'addition et en remarquant qu'en coordonnées affines $2y_3 = \lambda(x_1 + x_2 - 2x_3) - (y_1 + y_2)$ (avec les notations de III.9), Brier et Joye obtiennent une formule utilisant 13 multiplications et 5 carrés. Voici cette formule : soient $P_1 := (X_1 : Y_1 : Z_1)$ et $P_2 := (X_2 : Y_2 : Z_2)$ deux points de $E_{p^{s+1}} \setminus E_1$. On note $P_3 := (X_3 : Y_3 : Z_3)$, le point de $E_{p^{s+1}}$ tel que $P_3 = P_1 + P_2$. On a alors

$$\begin{array}{rcl} X_3 & = & 2FW \\ Y_3 & = & R(G - 2W) - L^2 \\ Z_3 & = & 2F^3 \end{array} \qquad \text{(III.13)}$$

avec $U_1 := X_1Z_2, U_2 := X_2Z_1, S_1 := Y_1Z_2, S_2 := Y_2Z_1, Z := Z_1Z_2, T := U_1 + U_2, M := S_1 + S_2, R := T^2 - U_1U_2 + aZ^2, F := ZM, L := MF, G := TL, W := R^2 - G$.

Addition de points de E_1 On utilise la formule du groupe formel. En posant $z := X/Y$, suivant les valeurs de s, on peut adapter les formules de loi du groupe formel (III.6) pour obtenir un algorithme d'addition en coordonnées projectives sans effectuer d'inversion.

Formules en coordonnées affines (2)

On utilise les formules obtenues en coordonnées projectives pour traiter les cas manquants en coordonnées affines.

Addition d'un point fini et d'un point au-dessus l'infini On donne ici une formule plus générale que (III.11) mais moins performante pour s grand. Soient $P_1 := (x_1, y_1)$ un point fini de $E_{p^{s+1}}$ et $\mathcal{O}_{k_2} = (k_2p : 1 : w(k_2p))$ un point de E_1 à additionner où k_2 désigne un élément de $\mathbf{Z}/p^s\mathbf{Z}$. En utilisant la formule d'addition en coordonnées projectives (III.12), et en convertissant le résultat en coordonnées affines, on trouve que le résultat de cette addition est $P_3 := (x_3, y_3)$ avec

$$\begin{aligned} x_3 &= (\lambda x_1 + \mu z_2)\gamma \\ y_3 &= -(\lambda y_1 + \mu)\gamma \end{aligned} \tag{III.14}$$

où $z_2 := k_2p$, $w := w(z_2)$, $\lambda := 1 - 3x_1z_2^2 + 2y_1w - 2az_2w - (ax_1 + 3b)w^2$, $\mu := -2y_1 + (3x_1^2 + a)z_2 - (y_1^2 - 2ax_1 - 3b)w$ et $\gamma := 1/(\lambda + \mu w)$.

Si $s = 1$, on trouve les formules plus simples :

$$\begin{aligned} x_3 &\equiv (x_1 - 2y_1k_2p) \pmod{p^2} \\ y_3 &\equiv \left(y_1 - (3x_1^2 + a)k_2p\right) \pmod{p^2} \end{aligned}$$

Addition de deux points finis au-dessus de deux points opposés Soient $P_1 := (x_1, y_1)$ et $P_2 := (x_2, y_2)$ deux points de $E_{p^{s+1}}$ tel que $\widetilde{P}_1 = -\widetilde{P}_2$ dans E_p. Le résultat étant un point de E_1, on le note $\mathcal{O}_{k_3}$, avec $k_3 \in \mathbf{Z}/p^s\mathbf{Z}$. Deux cas se présentent : $\widetilde{P}_1 \neq \widetilde{P}_2$ et $\widetilde{P}_1 = \widetilde{P}_2$, *i.e.*, les points réduits modulo p sont ou non d'ordre deux.

Cas où $\widetilde{P}_1$ n'est pas d'ordre deux dans E_p Pour trouver le résultat, on utilise la formule d'addition en coordonnées projectives de points distincts modulo p. On obtient

$$z_3 := k_3p = (\lambda x_1 + \mu x_2)\gamma$$

avec $\lambda = -x_1(3x_2^2 + a) + 2y_1y_2 + y_2^2 - 2ax_2 - 3b$, $\mu = x_2(3x_1^2 + a) - 2y_1y_2 - y_1^2 + 2ax_1 + 3b$ et $\gamma = -1/(\lambda y_1 + \mu y_2)$.

Comme $\widetilde{P}_1 = -\widetilde{P}_2$, on peut simplifier ces formules en tirant parti du fait que l'on travaille modulo p^{s+1}. On note $x_2 = x_1 + k_xp$ avec $k_x \in \mathbf{Z}/p^s\mathbf{Z}$. On note également

$y_2 = -y_1 + k_y p$ où k_y est un élément de $\mathbf{Z}/p^s\mathbf{Z}$. Cet élément est totalement déterminé par k_x et $\widetilde{\mathrm{P}}_1$.

En effet, par le lemme de Hensel, il n'existe qu'un point au-dessus de $-\widetilde{\mathrm{P}}_1$ d'abscisse x_2. Pour $s = 1$, on trouve que

$$k_y \equiv -\frac{3x_1^2 + a}{2y_1} k_x \pmod{p}$$

où y_1 est bien inversible car $\widetilde{\mathrm{P}}_1$ n'est pas d'ordre deux. On obtient finalement que

$$k_3 \equiv \frac{k_x}{2y_1} \pmod{p}.$$

Pour $s > 1$, les calculs deviennent vite inextricables et ne simplifient pas les formules données précédemment.

Cas où $\widetilde{\mathrm{P}}_1$ est d'ordre deux dans E_p Comme $\widetilde{\mathrm{P}}_1 = \widetilde{\mathrm{P}}_2$, on utilise la formule de double généralisé en projectives (III.13). On trouve que

$$z_3 := k_3 p = 2\gamma \mathrm{M}^3,$$

avec $\mathrm{T} := x_1 + x_2$, $\mathrm{M} := y_1 + y_2$, $\mathrm{R} := \mathrm{T}^2 - x_1 x_2 + a$, $\mathrm{L} := \mathrm{M}^2$, $\mathrm{G} := \mathrm{TL}$, et enfin, $\gamma := (\mathrm{R}(3\mathrm{G} - 2\mathrm{R}^2) - \mathrm{L}^2)^{-1}$.

Comme dans le cas précédent, ces formules peuvent se simplifier. Comme $\widetilde{\mathrm{P}}_1$ est d'ordre deux et comme $\widetilde{\mathrm{P}}_1 = \widetilde{\mathrm{P}}_2$, on a en fait $x_2 = x_1$ et il existe deux éléments m_1 et m_2 de $\mathbf{Z}/p^s\mathbf{Z}$ tel que $y_1 \equiv m_1 p \pmod{p^{s+1}}$ et $y_2 \equiv m_2 p \pmod{p^{s+1}}$.

Pour $s = 1$, on trouve que le résultat est $\mathcal{O}_{k_3}$, avec

$$k_3 \equiv -\frac{m_1 + m_2}{3x_1^2 + a} \pmod{p}$$

où $(3x_1^2 + a) \not\equiv 0 \pmod{p}$ car le point $\widetilde{\mathrm{P}}_1$ est non singulier.

Algorithme complet d'addition en coordonnées affines dans E_{p^2}

Soit P_1 et P_2 à additionner dans $\mathrm{E}_{p^2}(a, b)$. On suppose que les points finis sont représentés par des couples d'éléments (x_i, y_i) de $\mathbf{Z}/p^2\mathbf{Z} \times \mathbf{Z}/p^2\mathbf{Z}$, pour $i \in \{1, 2\}$. On suppose que les points au-dessus de l'infini sont notés sous la forme $\mathcal{O}_k$, où $k \in \mathbf{Z}/p\mathbf{Z}$. Les calculs sont effectués dans $\mathbf{Z}/p^2\mathbf{Z}$, sauf contre-indications. Si y est un élément de $\mathbf{Z}/p^2\mathbf{Z}$ tel que $y \equiv 0 \pmod{p}$, y/p désigne l'élément de $\mathbf{Z}/p\mathbf{Z}$ tel que $y \equiv (y/p)p \pmod{p^2}$.

On donne l'algorithme d'addition dans la figure III.1, page suivante.

Addition de P_1 et P_2 dans $\mathrm{E}_{p^2}(a,b)$

si P_1 *n'est pas fini* **alors**
  si P_2 *n'est pas fini* **alors** /* $\mathrm{P}_1 = \mathcal{O}_{k_1}$ et $\mathrm{P}_2 = \mathcal{O}_{k_2}$ */
    retourner : $\mathcal{O}_{k_1+k_2}$
  sinon /* $\mathrm{P}_1 = \mathcal{O}_{k_1}$ et P_2 est fini */
    retourner : $\left(x_2 - 2y_2k_1p,\ y_2 - (3x_2^2+a)k_1p\right)$

si P_2 *n'est pas fini* **alors** /* P_1 est fini et $\mathrm{P}_2 = \mathcal{O}_{k_2}$ */
  retourner : $\left(x_1 - 2y_1k_2p,\ y_1 - (3x_1^2+a)k_2p\right)$

si $x_1 \equiv x_2 \pmod p$ **alors**
  si $y_1 \equiv 0 \pmod p$ **alors** /* $\tilde{\mathrm{P}}_1 = -\tilde{\mathrm{P}}_2 \neq \mathcal{O}$ et $2\tilde{\mathrm{P}}_1 = \mathcal{O}$ */
    retourner : $\mathcal{O}_k$ avec $k \equiv -\frac{(y_1+y_2)/p}{3x_1^2+a} \pmod p$
  sinon si $y_1 \equiv -y_2 \pmod p$ **alors** /* $\tilde{\mathrm{P}}_1 = -\tilde{\mathrm{P}}_2 \neq \mathcal{O}$ et $2\tilde{\mathrm{P}}_1 \neq \mathcal{O}$ */
    retourner : $\mathcal{O}_k$ avec $k \equiv \frac{(x_2-x_1)/p}{2y_1} \pmod p$
  sinon /* $\tilde{\mathrm{P}}_1 = \tilde{\mathrm{P}}_2 \neq \mathcal{O}$ et $\tilde{\mathrm{P}}_1 \neq -\tilde{\mathrm{P}}_2$ */
    $\lambda := \frac{3x_1x_2+a}{y_2+y_1}$
sinon /* P_1 et P_2 sont finis et $\tilde{\mathrm{P}}_1 \neq \pm\tilde{\mathrm{P}}_2$ */
  $\lambda := \frac{y_2-y_1}{x_2-x_1}$

$x_3 := \lambda^2 - x_1 - x_2$

retourner : $\left(x_3, \lambda(x_1-x_3) - x_1\right)$

FIG. III.1 – Algorithme d'addition dans $\mathrm{E}_{p^2}(a,b)$

2.5. Un second cas particulier : $\mathrm{E}_{n^{s+1}}$ où n est un entier RSA et $s \geqslant 1$

C'est ce type de courbe que l'on va principalement utiliser au chapitre IV. On note $n = pq$ et on suppose que a et b sont deux éléments de $\mathbf{Z}/n^{s+1}\mathbf{Z}$ tels que $4a^3 + 27b^2$ soit inversible. D'après l'isomorphisme (III.2), le groupe $\mathrm{E}_{n^{s+1}}(a,b)$ est alors le produit des groupes précédemment étudiés $\mathrm{E}_{p^{s+1}}(a,b)$ et $\mathrm{E}_{q^{s+1}}(a,b)$. D'après la formule (III.7), l'ordre de $\mathrm{E}_{n^{s+1}}$ est $n^s\,|\mathrm{E}_n|$.

Travailler dans $\mathrm{E}_{n^{s+1}}$ sans connaître la factorisation de n conduit à une situation similaire à ce qu'on a vu pour E_n, en sous-section 2.2, à un point près, celui des points à l'infini. On note E_1 le sous-groupe de $\mathrm{E}_{n^{s+1}}$ constitué par les éléments au-dessus du point à l'infini de E_n. Par l'isomorphisme (III.2), E_1 est le produit de $\mathrm{E}_{1,p}$ et de $\mathrm{E}_{1,q}$ représentant respectivement le sous-groupe de $\mathrm{E}_{p^{s+1}}$, constitué par les points au-dessus de l'infini de E_p, et le sous-groupe de $\mathrm{E}_{q^{s+1}}$ des points au-dessus de l'infini de E_q. On a établi, page 42, le cardinal et une loi de groupe explicite pour ces deux derniers sous-groupes.

Par l'isomorphisme (III.2), on transfère ces propriétés sur E_1. Ainsi, E_1 est d'ordre n^s et ses éléments sont de la forme $(z : 1 : w(z))$ où z est un élément de $n(\mathbf{Z}/n^{s+1}\mathbf{Z})$ et où la série w est donnée par la formule (III.5). L'addition dans E_1 se fait par la loi du groupe formel F (cf. (III.6)).

Dans la pratique, on travaillera implicitement dans l'ensemble $\mathrm{E}'_{n^{s+1}}$ qui désigne l'ensemble des points (x, y) à coordonnées dans $\mathbf{Z}/n^{s+1}\mathbf{Z}$, satisfaisant dans $\mathbf{Z}/n^{s+1}\mathbf{Z}$ l'équation

$$y^2 = x^3 + ax + b,$$

auquel on rajoute les points de E_1. Comme dans la sous-section 2.2, on exclut les points semi-infinis, *i.e.*, les couples de la forme

$$(\mathrm{P}_p, \mathcal{O}_{k_p}) \quad \text{et} \quad (\mathcal{O}_{k_q}, \mathrm{P}_q)$$

où P_p et P_q sont des points finis respectifs de E_p et E_q, et $\mathcal{O}_{k_p}$ et $\mathcal{O}_{k_q}$ des points respectifs de $\mathrm{E}_{1,p}$ et $\mathrm{E}_{1,q}$.

Pour additionner des points de $\mathrm{E}'_{n^{s+1}}$, on utilisera directement dans $\mathbf{Z}/n^{s+1}\mathbf{Z}$ les formules d'additions vues en sous-section 2.4. Comme dans le cas $s = 0$ on n'obtiendra pas une loi de groupe, mais les cas fautifs seront atteints avec une probabilité négligeable car ils permettraient de factoriser n.

3. Anneaux d'entiers d'un corps quadratique modulo n

Tout comme les courbes elliptiques, les quotients de corps quadratiques ont joué un rôle pour la primalité (pseudo-premiers de Lucas, cf. [BSSW80]) et la factorisation (méthode $p+1$, cf. [Wil82]). Par contre, si les courbes elliptiques ont été utilisées massivement pour la cryptographie, les corps quadratiques ne l'ont été que rarement. On peut tout de même citer le cryptosystème LUC proposé par Smith et Lennon dans [SL93].

Dans toutes ces références et dans de nombreux autres articles sur le même sujet, la structure algébrique est cachée derrière l'utilisation des suites de Lucas. Dans ce chapitre, on va introduire des groupes finis très proches des quotients de $\mathbf{Z}$. Il s'agit de groupes d'éléments de norme 1 d'anneaux d'entiers de corps quadratiques modulo n. On verra que les suites de Lucas permettent en fait de calculer de manière très efficace l'exponentiation des éléments de ces groupes.

Muni de cette structure algébrique, on verra dans le chapitre IV que le cryptosystème LUC se décrit très naturellement comme une adaptation du cryptosystème RSA dans les quotients de corps quadratiques. On verra également comment cette structure permet d'utiliser les fonctions trappe décrites dans le chapitre II pour donner des systèmes plus rapides qu'en utilisant les courbes elliptiques.

3.1. Définitions

Notations

Soit Δ un entier non carré dans $\mathbf{Z}$. On note $\mathcal{O}_\Delta$ l'anneau des entiers de $\mathbf{Q}(\sqrt{\Delta})$. On suppose que l'entier n est premier avec Δ. Le quotient $\mathcal{O}_\Delta/n\mathcal{O}_\Delta$ étant un module libre de rang 2 sur $\mathbf{Z}/n\mathbf{Z}$, on peut définir l'application norme, notée $\mathrm{N}_{\mathcal{O}_\Delta/n\mathcal{O}_\Delta}$ (resp. l'application trace, notée $\mathrm{Tr}_{\mathcal{O}_\Delta/n\mathcal{O}_\Delta}$) de $\mathcal{O}_\Delta/n\mathcal{O}_\Delta$ sur $\mathbf{Z}/n\mathbf{Z}$, en associant à tout élément α de $\mathcal{O}_\Delta/n\mathcal{O}_\Delta$ le déterminant (resp. la trace) de l'endomorphisme de $(\mathbf{Z}/n\mathbf{Z})-$module, $\beta \mapsto \alpha\beta$.

Si $\mathrm{N}_{\mathcal{O}_\Delta}$ (resp. $\mathrm{Tr}_{\mathcal{O}_\Delta}$) est la norme usuelle (resp. la trace usuelle) de $\mathcal{O}_\Delta$ sur $\mathbf{Z}$, on a les relations suivantes :

$$\forall \alpha \in \mathcal{O}_\Delta,\ \mathrm{N}_{\mathcal{O}_\Delta/n\mathcal{O}_\Delta}\big(\Pi(\alpha)\big) \ \equiv\ \mathrm{N}_{\mathcal{O}_\Delta}(\alpha) \pmod{n},$$

$$\forall \alpha \in \mathcal{O}_\Delta,\ \mathrm{Tr}_{\mathcal{O}_\Delta/n\mathcal{O}_\Delta}\big(\Pi(\alpha)\big) \ \equiv\ \mathrm{Tr}_{\mathcal{O}_\Delta}(\alpha) \pmod{n},$$

où Π désigne la surjection canonique $\mathcal{O}_\Delta \to \mathcal{O}_\Delta/n\mathcal{O}_\Delta$. Vu ces relations on notera plus simplement, s'il n'y a pas de confusions possibles, respectivement N et Tr les applications norme et trace.

Comme la norme est un morphisme du groupe $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\times$ dans $(\mathbf{Z}/n\mathbf{Z})^\times$, l'ensemble des éléments de norme 1 de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\times$ forme un sous-groupe multiplicatif que l'on notera $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$.

Interprétation géométrique

Le groupe $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ peut aussi être défini comme l'ensemble des points finis de la conique d'équation affine

$$\mathrm{X}^2 - \Delta \mathrm{Y}^2 = 1,$$

sur l'anneau $\mathbf{Z}/n\mathbf{Z}$. Ceci est une conséquence du fait suivant : si α est un élément de $\mathcal{O}_\Delta/n\mathcal{O}_\Delta$, il existe un couple (x, y) d'éléments de $\mathbf{Z}/n\mathbf{Z}$ tel que $\alpha = x + y\sqrt{\Delta}$ et $\mathrm{N}(\alpha) = x^2 - \Delta y^2$.

La loi de groupe naturelle sur $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$ héritée de la multiplication dans $\mathbf{Q}$ a une interprétation géométrique. Pour voir cela, on se place sur le corps des réels. Sur la conique, l'élément neutre est le point de coordonnée $\mathrm{E} := (1, 0)$.

Soient A et B deux points distincts de la conique. On note D la droite parallèle à (AB) passant par E. Le second point d'intersection de D et de la conique est la somme de A et B (cf. figure III.2, dans le cas $\Delta < 0$, *i.e.*, le cas d'une ellipse). Si $\mathrm{A} = \mathrm{B}$, on prend la tangente à la conique en A au lieu de la droite (AB). On peut trouver dans le premier chapitre de [PS97] une étude plus approfondie de cette interprétation.

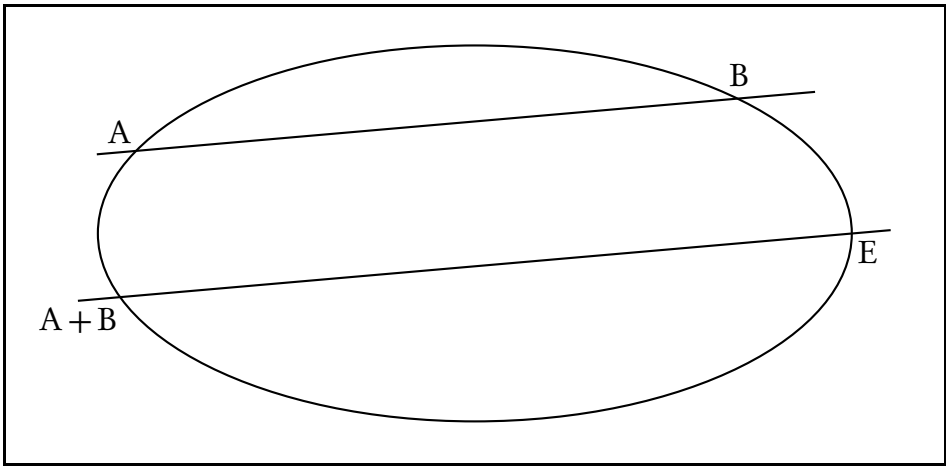

FIG. III.2 – Addition dans une conique

Structure de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$

Si p est un premier impair et x un entier tel que $p \nmid x$, on notera (x/p) le symbole de Legendre, défini par

$$\left(\frac{x}{p}\right) := \begin{cases} 1 & \text{si } x \text{ est un résidu quadratique modulo p,} \\ -1 & \text{si } x \text{ n'est pas un résidu quadratique modulo p.} \end{cases}$$

L'ordre $\varphi_\Delta(n)$ de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ est donné par le théorème suivant.

Théorème III – 7. *Soit Δ un entier non carré dans* **Z**.

Soit p un premier impair ne divisant pas Δ. Pour tout entier naturel s non nul,

$$\varphi_\Delta(p^s) = p^{s-1}\left(p - \left(\frac{\Delta}{p}\right)\right).$$

De plus, le groupe $(\mathcal{O}_\Delta/p^s\mathcal{O}_\Delta)^\wedge$ est cyclique.

Si n est un entier naturel impair, premier avec Δ, se décomposant en produits de nombres premiers p_i distincts :

$$n = \prod_{i\in I} p_i^{\alpha_i},$$

où les α_i, avec i élément de I, *sont tous des entiers naturels non nuls, alors on a l'isomorphisme de groupe :*

$$(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge \xrightarrow{\sim} \prod_{i\in I}\left(\mathcal{O}_\Delta/p_i^{\alpha_i}\mathcal{O}_\Delta\right)^\wedge.$$

Par conséquent, l'ordre de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ est $\varphi_\Delta(n) = \prod_{i\in I}\varphi_\Delta(p_i^{\alpha_i})$.

Le second point du théorème découle du théorème des restes chinois. On peut trouver une démonstration du premier point dans [Arn88], théorème 4.3.1 page 35.

Le groupe $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ a donc une structure très proche de celle de $\mathbf{Z}/n\mathbf{Z}$. Ces similarités permettront d'adapter facilement dans les quotients de corps quadratiques les cryptosystèmes existants dans les quotients de $\mathbf{Z}$. Dans la sous-section suivante, on s'intéresse aux suites de Lucas qui permettront de réaliser des exponentiations rapides dans le groupe $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. Ces suites seront un élément clef pour construire des systèmes efficaces.

3.2. Suites de Lucas

Certains éléments de cette sous-section proviennent de [Cas].

Définition

Soient P et Q deux entiers, tels que $\Delta := P^2 - 4Q$ ne soit pas un carré dans $\mathbf{Z}$. Les suites de Lucas sont données par des relations de récurrences linéaires à deux pas, avec des valeurs particulières pour les deux premiers termes. Pour tout entier $k \geqslant 1$,

$$\begin{aligned} U_{k+1}(P,Q) &= PU_k(P,Q) - QU_{k-1}(P,Q), \quad U_1(P,Q) = 1, \quad U_0(P,Q) = 0, \\ V_{k+1}(P,Q) &= PV_k(P,Q) - QV_{k-1}(P,Q), \quad V_1(P,Q) = P, \quad V_0(P,Q) = 2. \end{aligned}$$

On verra dans la suite le lien entre ces suites et l'exponentiation d'entiers quadratiques. On donne d'abord une méthode efficace pour calculer des termes de certaines suites de Lucas.

Calcul de $V_k(P,1)$ et de $U_k(P,1)$

Plusieurs algorithmes ont été proposés pour calculer les termes des suites de Lucas. Pour les applications au groupe $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$, on aura besoin de calculer seulement des termes des suites $(V_i(P,1))_{i\in\mathbf{N}}$ et $(U_i(P,1))_{i\in\mathbf{N}}$, *i.e.*, avec Q (le second paramètre des suites de Lucas), égal à 1. On verra pourquoi dans le paragraphe suivant faisant le lien entre ces suites et l'exponentiation des éléments de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. Ceci permettra un gain calculatoire notable.

On utilise ici une version simplifiée (en prenant $Q = 1$) de l'algorithme présenté dans le cadre général dans [JQ96].

Soit k un entier, on veut calculer les termes $V_k(P,1)$ et $U_k(P,1)$ où $P^2 - 4$ n'est pas un carré. Pour cela, on écrit la décomposition binaire de k : $k_0 k_1 \dots k_{m-1}$, le bit de poids fort étant k_0 et $m := |k|_2$. On va utiliser un algorithme du type "left to right". On suppose $m \geqslant 2$ et on note $f_0 := k_0 = 1$, puis pour $i = 1, \dots, m-1$, $f_i = 2f_{i-1} + k_i$, de telle sorte que $f_{m-1} = k$. Pour alléger les notations, pour tout entier naturel i, on notera V_i (resp. U_i) pour $V_i(P,1)$ (resp. pour $U_i(P,1)$). On va utiliser deux des nombreuses formules vérifiées par les suites de Lucas :

$$\forall i \geqslant j \geqslant 0, \quad V_{i+j} = V_i V_j - V_{i-j} \tag{III.15}$$

$$\forall i \geqslant j \geqslant 0, \quad U_{i+j} = U_i V_j - U_{i-j} \tag{III.16}$$

Calcul de $\mathrm{V}_k(\mathrm{P},1)$ On suppose k impair dans un premier temps. De la formule (III.15), on tire

$$\begin{aligned} \forall j \in \mathbf{N}, \quad \mathrm{V}_{2j} &= \mathrm{V}_j^2 - 2, \\ \forall j \in \mathbf{N}, \quad \mathrm{V}_{2j+1} &= \mathrm{V}_{j+1}\mathrm{V}_j - \mathrm{P}. \end{aligned}$$

On a $\mathrm{V}_{f_0} = \mathrm{V}_1 = \mathrm{P}$ et $\mathrm{V}_{f_0+1} = \mathrm{V}_2 = \mathrm{P}^2 - 2$. Supposons que pour un entier i, tel que $1 \leqslant i \leqslant m-2$, on ait calculé $\mathrm{V}_{f_{i-1}}$ et $\mathrm{V}_{f_{i-1}+1}$. On calcule les termes suivants par

$$\text{Si } k_i = 0, \begin{cases} \mathrm{V}_{f_i} = \mathrm{V}_{2f_{i-1}} = \mathrm{V}_{f_{i-1}}^2 - 2, \\ \mathrm{V}_{f_i+1} = \mathrm{V}_{2f_{i-1}+1} = \mathrm{V}_{f_{i-1}+1}\mathrm{V}_{f_{i-1}} - \mathrm{P}. \end{cases}$$

$$\text{Si } k_i = 1, \begin{cases} \mathrm{V}_{f_i} = \mathrm{V}_{2f_{i-1}+1} = \mathrm{V}_{f_{i-1}+1}\mathrm{V}_{f_{i-1}} - \mathrm{P}, \\ \mathrm{V}_{f_i+1} = \mathrm{V}_{2(f_{i-1}+1)} = \mathrm{V}_{f_{i-1}+1}^2 - 2. \end{cases}$$

Au dernier rang, pour $i = m-1$, il suffit de calculer $\mathrm{V}_{f_i} = \mathrm{V}_{f_{m-1}} = \mathrm{V}_k$. Comme k est supposé impair, on a $k_{m-1} = 1$ et

$$\mathrm{V}_k = \mathrm{V}_{2f_{m-2}+1} = \mathrm{V}_{f_{l-2}+1}\mathrm{V}_{f_{m-2}} - \mathrm{P}.$$

Notons que le nombre et la nature des opérations effectuées sont indépendants des bits de k. L'algorithme pour calculer V_k, quand k est impair, n'est donc pas atteint par les attaques de type SPA (pour "Simple Power Analysis").

Si k est pair, on l'écrit sous la forme $k =: 2^s k'$ où $s \in \mathbf{N}$ et k' est impair. On calcule $\mathrm{V}_{k'}$ comme vu plus haut. Soit $j \in \mathbf{N}$ tel que $1 \leqslant j \leqslant s$, si $\mathrm{V}_{2^{j-1}k'}$ est connu, on a

$$\mathrm{V}_{2^j k'} = \mathrm{V}_{2^{j-1}k'}^2 - 2.$$

On obtient ainsi V_k au bout de s itérations. Au final, pour obtenir $\mathrm{V}_k(\mathrm{P},1)$ avec $k = 2^s k'$ où k' est impair, on utilise

$$\left(2\left|k'\right|_2 + s - 2\right)$$

multiplications dans $\mathbf{Z}$ dès que $|k'|_2 \geqslant 2$. L'algorithme pour calculer V_k est résumé dans la figure III.3.

Calcul de $\mathrm{U}_k(\mathrm{P},1)$ On ne peut pas faire mieux que de calculer U_k en même temps que V_k. On suppose toujours que $|k|_2 \geqslant 2$ et on garde les notations du paragraphe précèdent. On suppose, dans un premier temps, que k est impair.

De la formule (III.16), on tire

$$\begin{aligned} \forall j \in \mathbf{N}, \quad \mathrm{U}_{2j} &= \mathrm{U}_j\mathrm{V}_j, \\ \forall j \in \mathbf{N}, \quad \mathrm{U}_{2j+1} &= \mathrm{U}_{j+1}\mathrm{V}_j - 1. \end{aligned}$$

Si l'on calcule simultanément V_{f_i} et V_{f_i+1}, il suffit de calculer U_{f_i+1} pour $i = 0, \ldots, m-1$. En effet, pour $i = 0$, on a $f_0 = 1$ et $\mathrm{U}_{f_0+1} = \mathrm{U}_2 = \mathrm{P}$. Supposons que $\mathrm{U}_{f_{i-1}+1}$, $\mathrm{V}_{f_{i-1}}$ et $\mathrm{V}_{f_{i-1}+1}$ soient connus pour un i tel que $1 \leqslant i \leqslant m-2$, on a alors

Calcul de $V_k(P,1)$

entrée : $P \in \mathbf{Z}$ tel que P^2-4 n'est pas un carré et $k>0$ donné sous la forme $k = 2^s \sum_{i=0}^{m-1} k_i 2^{m-1-i}$ avec $k_0 = k_{m-1} = 1$.

$V_\ell := 2, V_h := P$.

pour $i = 0, \ldots, m-2$ **faire**
  si $k_i = 0$ **alors**
    $V_h := V_\ell V_h - P, V_\ell := V_\ell^2 - 2$,
  sinon
    $V_\ell := V_\ell V_h - P, V_h := V_h^2 - 2$.
  fin
fin

$V_\ell := V_\ell V_h - P$.

pour $i = 1, \ldots, s$ **faire**
  $V_\ell := V_\ell^2 - 2$.
fin

sortie : V_ℓ

FIG. III.3 – Algorithme de calcul de $V_k(P,1)$

$$\begin{array}{rrcccl} \text{Si } k_i = 0, & U_{f_i+1} & = & U_{2f_{i-1}+1} & = & U_{f_{i-1}+1}V_{f_{i-1}} - 1, \\ \text{et, si } k_i = 1, & U_{f_i+1} & = & U_{2(f_{i-1}+1)} & = & U_{f_{i-1}+1}V_{f_{i-1}+1}. \end{array}$$

Au dernier rang, comme k est impair, on a $k_{m-1} = 1$. On déduit donc U_k de $U_{f_{m-2}+1}$ de la façon suivante :

$$U_k = U_{f_{m-1}} = U_{2f_{m-2}+1} = U_{f_{m-2}+1}V_{f_{m-2}} - 1.$$

Par contre, si on avait supposé k pair, on aurait eu besoin de la valeur de $U_{f_{m-2}}$.

Supposons maintenant que k est pair. On écrit k sous la forme $k =: 2^s k'$ où $s \in \mathbf{N}$ et k' est impair. On calcule $V_{k'}$ et $U_{k'}$ comme vu plus haut. Supposons les $V_{2^j k'}$ connus pour tout entier j entre 1 et s. Si $U_{2^{j-1}k'}$ est connu pour un entier j entre 1 et $s-1$, on a

$$U_{2^j k'} = U_{2^{j-1}k'}V_{2^{j-1}k'}.$$

On obtient ainsi U_k au bout de s itérations.

Au final, obtenir le couple $\left(V_k(P,1), U_k(P,1)\right)$ avec $k = 2^s k'$, k' impair, se fait en

$$\left(3\left|k'\right|_2 + 2s - 3\right)$$

multiplications dans $\mathbf{Z}$ si $|k'|_2 \geqslant 2$. L'algorithme pour faire ce calcul est donné dans la figure III.4, page suivante.

Notons que si l'on désire avoir les valeurs $V_k \bmod n$ et $U_k \bmod n$, on utilisera exactement les mêmes algorithmes en remplaçant les opérations dans $\mathbf{Z}$ par les mêmes opérations dans $\mathbf{Z}/n\mathbf{Z}$.

Calcul de $V_k(P,1)$ et $U_k(P,1)$

entrée : $P \in \mathbf{Z}$ tel que P^2-4 n'est pas un carré et $k>0$ donné sous la forme $k = 2^s \sum_{i=0}^{m-1} k_i 2^{m-1-i}$ avec $k_0 = k_{m-1} = 1$.

$V_\ell := 2, V_h := P$, $U := 1$.

pour $i = 0, \ldots, m-2$ **faire**
  si $k_i = 0$ **alors**
    $U := UV_\ell - 1, V_h := V_\ell V_h - P, V_\ell := V_\ell^2 - 2,$
  sinon
    $U := UV_h, V_\ell := V_\ell V_h - P, V_h := V_h^2 - 2,.$
  fin
fin

$U := UV_\ell - 1, V_\ell := V_\ell V_h - P.$

pour $i = 1, \ldots, s$ **faire**
  $U := UV_\ell, V_\ell := V_\ell^2 - 2.$
fin

sortie : (U, V_ℓ)

FIG. III.4 – Algorithme de calcul de $V_k(P,1)$ et $U_k(P,1)$

Signalons l'existence d'autres algorithmes pour calculer V_k et U_k. L'algorithme de la figure III.3 est étudié dans [YL95]. On y analyse aussi un algorithme commençant par le bit le plus faible, de même complexité, mais nécessitant une variable supplémentaire (il faut aussi calculer $V_{2^{i+1}-f_i}$). Dans des articles antérieurs : [SL93, Wil82], on utilisait un algorithme "left to right" calculant au rang i, V_{f_i} et V_{f_i-1}, au lieu de V_{f_i+1} ce qui rajoute en moyenne $|k|_2/2$ multiplications.

Suites de Lucas et exponentiation

On fait maintenant le lien entre les suites de Lucas et le groupe $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. On se replace dans le cas général : P et Q sont deux entiers, tels que $\Delta := P^2 - 4Q$ ne soit pas un carré.

On note

$$\alpha := \frac{P + \sqrt{\Delta}}{2},$$

une des racines du polynôme $X^2 - PX + Q$ de $\mathbf{Z}[X]$. On définit ainsi un élément de l'anneau

des entiers $\mathcal{O}_\Delta$ du corps $\mathbf{Q}(\sqrt{\Delta})$. On a alors

$$\forall k \in \mathbf{N}, \quad \alpha^k = \frac{V_k(P,Q) + U_k(P,Q)\sqrt{\Delta}}{2},$$

dans l'anneau $\mathcal{O}_\Delta$.

On en déduit que

$$\forall k \in \mathbf{N}, \quad V_k(P,Q) = \mathrm{Tr}(\alpha^k). \tag{III.17}$$

On peut généraliser ce résultat pour calculer des puissances d'éléments de l'anneau $\mathcal{O}_\Delta$ avec des Δ non carrés arbitraires. On peut faire de même dans $\mathcal{O}_\Delta/n\mathcal{O}_\Delta$ avec Δ et n premier entre eux.

Proposition III – 8. *Soit Δ un entier non carré dans $\mathbf{Z}$ et n un entier impair tel que Δ et n soient premiers entre eux. Soit α un élément de $\mathcal{O}_\Delta$ et x, y deux entiers tels que $\alpha \equiv x + y\sqrt{\Delta}$ $(\mathrm{mod}\ n\mathcal{O}_\Delta)$. Pour tout entier naturel k, on a*

$$\begin{aligned} \alpha^k &\equiv \frac{V_k\big(2x, N(\alpha)\big)}{2} + yU_k\big(2x, N(\alpha)\big)\sqrt{\Delta} \pmod{n\mathcal{O}_\Delta}, \\ \mathrm{Tr}(\alpha^k) &\equiv V_k\big(2x, N(\alpha)\big) \pmod{n\mathcal{O}_\Delta}. \end{aligned}$$

Démonstration. Soit $P := 2x$ et $Q := N(\alpha) = x^2 - \Delta y^2$. L'entier Δ n'étant pas un carré, $P^2 - 4Q$ non plus. Le résultat est trivial pour $k = 0$ et $k = 1$. Pour tout entier k supérieur ou égal à 1, on voit facilement que

$$\alpha^{k+1} = P\alpha^k - Q\alpha^{k-1}.$$

On conclut par récurrence en utilisant cette égalité combinée avec les relations de récurrences linéaires vérifiées par les suites de Lucas. □

Grâce à cette proposition on voit le lien entre l'exponentiation d'un entier quadratique et les suites de Lucas. Dans la suite, on ne va utiliser que des éléments α de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. Pour tout entier P, on a $V\big(P, N(\alpha)\big) \equiv V(P,1) \pmod{n}$, et $U\big(P, N(\alpha)\big) \equiv U(P,1) \pmod{n}$. Le second paramètre des suites de Lucas utilisées sera donc toujours égal à 1. On notera $V_k(P)$ (resp. $U_k(P)$) pour $V_k(P,1)$ (resp. $U_k(P,1)$). On voit ainsi pourquoi on a restreint l'exposé du calcul des suites de Lucas au cas $Q = 1$.

Pour finir sur les suites de Lucas, notons qu'elles apportent un réel gain pour calculer α^k avec α dans $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. L'exponentiation en utilisant la propriété III – 8 prend moins de $3|k|_2$ multiplications dans $\mathbf{Z}/n\mathbf{Z}$, en utilisant l'algorithme de la figure III.4.

En revanche, le calcul direct dans $\mathcal{O}_\Delta$ est plus coûteux. En effet, si $\alpha = \alpha_1 + \alpha_2\sqrt{\Delta}$ et $\beta = \beta_1 + \beta_2\sqrt{\Delta}$ sont deux éléments de $\mathcal{O}_\Delta$, le calcul de $\alpha\beta = \alpha_1\beta_1 + \Delta\alpha_2\beta_2 + (\alpha_1\beta_2 + \alpha_2\beta_1)\sqrt{\Delta}$ prend 5 multiplications. Celui de $\alpha^2 = \alpha_1^2 + \alpha_2^2\Delta + 2\alpha_1\alpha_2\sqrt{\Delta}$ en prend 4. Ainsi, on peut calculer α^k dans $\mathcal{O}_\Delta$ en utilisant un algorithme de type "square et multiply" en moyenne en $4|k|_2 + \frac{5}{2}|k|_2 = \frac{13}{2}|k|_2$ multiplications.

L'utilisation des suites de Lucas permet donc de gagner plus d'un facteur 2 pour l'exponentiation des éléments de norme 1.

3.3. Noyau de la réduction $\left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^\wedge \to (\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$

Dans cette sous-section, n sera un entier RSA et s un entier naturel non nul. On note $n = pq$ et on suppose toujours que n et Δ sont premiers entre eux et que Δ n'est pas un carré dans $\mathbf{Z}$.

On désigne par Π la surjection canonique de $\left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^\wedge$ sur $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. On se propose d'étudier le noyau de Π. Cette étude est à mettre en rapport avec celle faite du sous-groupe E_1 dans le cadre des courbes elliptiques (cf. pages 38 et suivante). Elle nous sera nécessaire pour utiliser les fonctions trappe introduites au chapitre II.

Générateur du noyau de la réduction

D'après l'étude faite dans [Arn88], section 4.3, comme p est premier, le morphisme

$$\left(\mathcal{O}_\Delta/p^{s+1}\mathcal{O}_\Delta\right)^\wedge \longrightarrow (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\wedge$$

est surjectif et son noyau est cyclique d'ordre p^s. De plus, dans le cas $s \geqslant 2$, un élément de $\left(\mathcal{O}_\Delta/p^{s+1}\mathcal{O}_\Delta\right)^\wedge$ de la forme $1+\gamma$, avec $\gamma \in p\mathcal{O} \setminus p^2\mathcal{O}$ sera un générateur de ce noyau.

Modulo q, on a le même résultat. Comme n est le produit pq avec p et q distincts, le théorèmes des restes chinois assure que ce résultat tient encore modulo n. C'est à dire que $\ker\Pi$ est cyclique d'ordre n^s et qu'un élément de $\ker\Pi$ de la forme $1+\gamma$, avec $\gamma \in n\mathcal{O}\setminus n^2\mathcal{O}$, sera un générateur de ce noyau dans le cas $s \geqslant 2$. On va construire explicitement un tel élément.

Dans le cas $s = 1$, $\alpha := 1 + n\sqrt{\Delta}$ est bien d'ordre n car pour tout entier positif k,

$$\alpha^k = 1 + kn\sqrt{\Delta},$$

i.e., $\alpha^k = 1$ si et seulement si $k \equiv 0 \pmod{n}$.

Pour des s supérieurs, il suffit de relever par récurrence cet élément en utilisant le lemme de Hensel. On utilise la proposition suivante.

Proposition III – 9 (Relèvement des éléments de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ dans $\left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^\wedge$)**.**

Soit $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ de discriminant Δ, avec $\gcd(n, 2\Delta) = 1$ *et α un élément de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. On cherche à relever α modulo n^{s+1}. On va procéder par récurrence en utilisant le lemme de Hensel. Supposons que l'on ait déjà relevé α dans $\left(\mathcal{O}_\Delta/n^k\mathcal{O}_\Delta\right)^\wedge$ pour un entier k tel que $1 \leqslant k \leqslant s$. On relève α modulo n^{k+1}. Pour cela, on écrit α sous la forme $\alpha_1 + \alpha_2\sqrt{\Delta}$ avec α_1 et α_2 deux entiers définis modulo n^k. On note f le polynôme*

$$f(x, y) := x^2 - \Delta y^2 - 1,$$

de telle sorte que $f(\alpha_1, \alpha_2) \equiv 0 \pmod{n^k}$. On relève cette racine par le lemme de Hensel. On a deux cas :

— *si* $\gcd(\alpha_2, n) = 1$, *on relève α dans $\left(\mathcal{O}_\Delta/n^{k+1}\mathcal{O}_\Delta\right)^\wedge$ en $\alpha' := \alpha_1 + \alpha_2'\sqrt{\Delta}$ avec*

$$\alpha_2' \equiv \alpha_2 + \left((2\Delta\alpha_2)^{-1} \bmod n\right) f(\alpha_1, \alpha_2) \pmod{n^{k+1}};$$

— *sinon, si* $\mathrm{pgcd}(\alpha_1, n) = 1$, *on relève* α *en* $\alpha' := \alpha'_1 + \alpha_2\sqrt{\Delta}$ *avec*

$$a' \equiv \alpha_1 - \left((2\alpha_1)^{-1} \bmod n\right) f(\alpha_1, \alpha_2) \pmod{n^{k+1}}.$$

Au bout de $(s-1)$ *étapes, on trouve un élément* $(\alpha)_l$ *de* $\left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^\wedge$ *tel que* $(\alpha)_l \equiv \alpha \pmod{n}$.

Remarque. Si n est un entier RSA, pour un élément α pris au hasard dans $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$, on est assuré d'être dans l'un ou l'autre des deux cas pour chaque itération du relèvement. En effet, n ne peut diviser les deux coordonnées puisque $0 \notin (\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. Si les coordonnées ne sont pas premières avec n, c'est donc que l'on a obtenu une factorisation de n.

En toute rigueur, si n est premier, on est dans au moins un des cas puisque 0 n'est pas un élément de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. Par contre, on peut trouver des cas pathologiques. Par exemple, si n est le produit de deux premiers : $n = pq$, et si l'on a $\Delta \equiv -1 \pmod{p}$. Il existe alors un élément α de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ tel que $\alpha \equiv \sqrt{\Delta} \pmod{p}$ et $\alpha \equiv 1 \pmod{q}$. En écrivant α sous la forme $\alpha_1 + \alpha_2\sqrt{\Delta}$, on a alors $\mathrm{pgcd}(\alpha_1, n) = p$ et $\mathrm{pgcd}(n, \alpha_2) = q$. Dans un tel cas, on applique la proposition modulo p et q et on obtient le relevé par utilisation des restes chinois.

Dans le cas de l'élément $\alpha = 1 + n\sqrt{\Delta}$, le procédé va fonctionner. On est alors dans le deuxième cas de la proposition. On relève α de $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$ dans $\left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^\wedge$ et on obtient :

$$\begin{aligned}\alpha \;=\; & n\sqrt{\Delta} + 1 + \tfrac{1}{2}\Delta n^2 - \tfrac{1}{2^3}\Delta^2 n^4 + \tfrac{1}{2^4}\Delta^3 n^6 - \tfrac{5}{2^7}\Delta^4 n^8 \\ & + \tfrac{7}{2^8}\Delta^5 n^{10} - \tfrac{21}{2^{10}}\Delta^6 n^{12} + \tfrac{33}{2^{11}}\Delta^7 n^{14} - \tfrac{429}{2^{15}}\Delta^8 n^{16} + \tfrac{715}{2^{16}}\Delta^9 n^{18} \ldots\end{aligned} \tag{III.18}$$

Cet élément α est alors un générateur du noyau de Π.

Calcul du logarithme en base α dans $\ker\Pi$

Pour adapter les fonctions trappe du chapitre II dans $\left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^\wedge$, on aura besoin de savoir calculer efficacement le logarithme en base α dans $\ker\Pi$, avec α défini en (III.18).

Soit k un entier défini modulo n^s. On écrit α^k sous la forme $\alpha_1 + \alpha_2\sqrt{\Delta}$, où α_1, α_2 sont des entiers définis modulo n^{s+1}. On trouve, par exemple modulo n^{10}, à partir de l'expression (III.18) de α que :

$$\begin{aligned}\gamma := \tfrac{\alpha_2}{n} \;\equiv\; & k \\ & + \tbinom{k}{2}\left(-\tfrac{5}{64}\Delta^4 n^8 + \tfrac{1}{8}\Delta^3 n^6 - \tfrac{1}{4}\Delta^2 n^4 + \Delta n^2\right) \\ & + \tbinom{k}{3}\left(\tfrac{15}{64}\Delta^4 n^8 - \tfrac{3}{8}\Delta^3 n^6 + \tfrac{3}{4}\Delta^2 n^4 + \Delta n^2\right) \\ & + \tbinom{k}{4}\left(-\tfrac{1}{8}\Delta^4 n^8 + 2\Delta^2 n^4\right) \\ & + \tbinom{k}{5}\left(-\tfrac{15}{16}\Delta^4 n^8 + \tfrac{5}{2}\Delta^3 n^6 + \Delta^2 n^4\right) \\ & + \tbinom{k}{6}\left(\tfrac{7}{4}\Delta^4 n^8 + 3\Delta^3 n^6\right) \\ & + \tbinom{k}{7}\left(\tfrac{21}{4}\Delta^4 n^8 + \Delta^3 n^6\right) \\ & + \tbinom{k}{8}\left(4\Delta^4 n^8\right) \\ & + \tbinom{k}{9}\left(\Delta^4 n^8\right) \pmod{n^{10}}\end{aligned} \tag{III.19}$$

Plus généralement, on voit que l'on a l'expression suivante :

$$\gamma \equiv k + \sum_{i=1}^{\lceil \frac{s}{2} \rceil - 1} \left[\binom{k}{2i} a_i + \binom{k}{2i+1} b_i \right] \pmod{n^s},$$

où $n^{2i} \mid a_i$ et $n^{2i} \mid b_i$, pour $i = 1, \ldots, \lceil \frac{s}{2} \rceil - 1$.

On note $k_j := k \bmod n^{2j}$. On récupère les k_j par récurrence pour $j = 1, \ldots, l$ avec $l := \lceil \frac{s}{2} \rceil - 1$. On trouvera ainsi l'entier $(k \bmod n^s)$. On a déjà $k_1 = \gamma \bmod n^2$. Soit j un entier tel que $1 \leqslant j \leqslant l$. Supposons connus les k_i pour $1 \leqslant i < j$, on a alors

$$\gamma \equiv k_j + \sum_{i=1}^{j-1} \left[\binom{k}{2i} a_i + \binom{k}{2i+1} b_i \right] \pmod{n^{2j}}. \tag{III.20}$$

Comme pour tout entier i avec $1 \leqslant i < j$, $n^{2i} \mid a_i$, on voit facilement, en écrivant les coefficients binomiaux sous formes développées, que

$$\binom{k}{2i} a_i \equiv \binom{k_{j-i}}{2i} a_i \pmod{n^{2j}},$$

et de même pour les b_i (notons que comme n est un entier RSA, les dénominateurs intervenant dans les coefficients binomiaux sont inversibles si $s < \min(p, q)$). Comme $1 \leqslant i < j$, on a bien $1 \leqslant j - i < j$. Les k_{j-i} sont donc connus et la formule (III.20) permet de calculer k_j.

Dans la pratique, on peut se servir uniquement de la valeur de k_{j-1}, cela donne un algorithme avec moins de stockage. Le calcul des coefficients binomiaux se fait par récurrence. On donne dans la figure III.5 l'algorithme correspondant. Le coût de cet algorithme est inférieur au coût de $s(s+2)/2$ multiplications modulo n^s.

3.4. Lien avec le tore algébrique

Dans cette sous-section, on étudie les liens entre le groupe $(\mathcal{O}_\Delta / n\mathcal{O}_\Delta)^\wedge$ et le tore algébrique.

L'utilisation du tore algébrique pour la cryptographie est due à Rubin et Silverberg (cf. [RS03]). Cette structure est plutôt destinée à être utilisée pour des cryptosystèmes basés sur le problème du logarithme discret alors que le groupe $(\mathcal{O}_\Delta / n\mathcal{O}_\Delta)^\wedge$ est destiné à être utilisé pour des cryptosystèmes basés sur le problème de la factorisation, comme on le verra au chapitre IV.

Si p est un nombre premier impair et Δ un entier non carré dans $\mathbf{Z}$ et non divisible par p, le groupe $(\mathcal{O}_\Delta / p\mathcal{O}_\Delta)^\wedge$ peut aussi être utilisé pour des problèmes de logarithme discret (cf. [SS94]). On va voir qu'on a un lien très étroit entre ce groupe et le tore algébrique T_2.

Dans un premier temps, on motive l'apparition du tore en cryptographie, puis on définit le tore algébrique et enfin on voit le lien avec le groupe $(\mathcal{O}_\Delta / p\mathcal{O}_\Delta)^\wedge$. Le matériel de cette étude est dérivé de [RS03].

Calcul du logarithme discret en base α dans $\ker\Pi$

entrée : n, s, Δ et $c \in \ker\Pi$, donné sous la forme (c_1, c_2) telle que
$c \equiv c_1 + c_2\sqrt{\Delta} \pmod{n^{s+1}}$.

pré-calculs : $l := \lceil \frac{s}{2} \rceil$, on calcule les a_i et b_i pour $1 \leqslant i < l$ (cf. III.19).

$\gamma := \frac{c_2}{n} \bmod n^s$,
$k := \gamma \bmod n^2$, /* $k = k_1$ */
pour $j = 2, \ldots, l$ **faire**
  $k_{\text{pré}} := k$, /* $k_{\text{pré}} = k_{j-1}$ */
  $k := \gamma \bmod n^{2j}$,
  $c := k_{\text{pré}}$, /* $c = \binom{k_{j-1}}{1}$ */
  si $2j > s$ **alors** $t := s$, **sinon** $t := 2j$. /* Si s est impair */
  pour $i = 1, \ldots, j-1$ **faire**
    $c := c \times \frac{k_{\text{pré}} - 2i + 1}{2i} \bmod n^t$, /* $c = \binom{k_{j-1}}{2i}$ */
    $k := k - c \times a_i \bmod n^t$,
    $c := c \times \frac{k_{\text{pré}} - 2i}{2i+1} \bmod n^t$, /* $c = \binom{k_{j-1}}{2i+1}$ */
    $k := k - c \times b_i \bmod n^t$. /* $k = k_j$ */
  fin
fin

sortie : k

FIG. III.5 – Algorithme de calcul du logarithme discret dans $\ker\Pi$

Pourquoi le tore ?

Soit $k = \mathbf{F}_q$ et soit $\mathrm{K} = \mathbf{F}_{q^m}$, une extension de corps de degré $m > 1$. On souhaite faire un système reposant sur le problème du logarithme discret dans $\mathrm{K}^\times$, *a priori* plus difficile que dans le corps de base.

Soit g un générateur de $\mathrm{K}^\times$ et h un élément quelconque de $\mathrm{K}^\times$. On cherche un entier s tel que $h = g^s$. L'exposant s est défini modulo $(q^m - 1)$.

Soit d un diviseur de m distinct de m, $\mathbf{F}_{q^d}$ est un sous-corps de K. On a une expression pratique de la norme $\mathrm{N}_{\mathrm{K}/\mathbf{F}_{q^d}}$:

$$\forall x \in \mathrm{K},\ \mathrm{N}_{\mathrm{K}/\mathbf{F}_{q^d}}(x) = x^{\frac{q^m-1}{q^d-1}},$$

où l'égalité a lieu dans $\mathbf{F}_{q^d}$. Ainsi, comme g est un générateur de $\mathrm{K}^\times$, $\mathrm{N}_{\mathrm{K}/\mathbf{F}_{q^d}}(g)$ sera un générateur de $\mathbf{F}_{q^d}^\times$. La norme permet donc d'avoir des informations sur le logarithme discret :

si $h = g^s$ dans $\mathrm{K}^\times$ on aura

$$\mathrm{N}_{\mathrm{K}/\mathbf{F}_{q^d}}(h) = \left(\mathrm{N}_{\mathrm{K}/\mathbf{F}_{q^d}}(g)\right)^s,$$

dans $\mathbf{F}_{q^d}$. Ainsi, si on sait résoudre le logarithme discret dans le sous-corps $\mathbf{F}_{q^d}$ on obtient s modulo $(q^d - 1)$.

Pour être sûr de garder la sécurité de K, il faut donc travailler dans un sous-groupe constitué d'éléments de norme 1 sur toutes les sous-extensions. C'est là qu'intervient le tore.

Définition

On donne la définition issue de l'article original de Rubin et Silverberg ([RS03], section 3).

Définition III – 10. *Soit $k = \mathbf{F}_q$ et $\mathrm{K} = \mathbf{F}_{q^m}$ une extension de degré $m > 1$. On définit $\mathrm{T}_m(\mathbf{F}_q)$ par*

$$\mathrm{T}_m(\mathbf{F}_q) = \left\{ x \in \mathrm{K}^\times,\ \forall 1 \leq d < m,\ d \mid m,\ \mathrm{N}_{\mathrm{K}/\mathrm{F}_{q^d}}(x) = 1 \right\}.$$

Il est immédiat que $\mathrm{T}_m(\mathbf{F}_q)$ est un sous-groupe de $\mathrm{K}^\times$ (c'est l'intersection des noyaux des normes sur toutes les sous-extensions). On note $\Phi_m(\mathrm{X})$ le m–ième polynôme cyclotomique. Le lemme suivant précise la nature de $\mathrm{T}_m(\mathbf{F}_q)$.

Lemme III – 11. *Avec les notations de la définition, $\mathrm{T}_m(\mathbf{F}_q)$ est le sous-groupe de $\mathrm{K}^\times$ d'ordre $\Phi_m(q)$. De plus, si $h \in \mathrm{T}_m(\mathbf{F}_q)$ est un élément ne divisant pas m alors h n'appartient à aucun sous-corps de* K.

Démonstration. Soit H le sous-groupe de $\mathrm{K}^\times$ d'ordre $\Phi_m(q)$. On note g un générateur de $\mathrm{K}^\times$ de telle sorte que $g^{\frac{q^m-1}{\Phi_n(q)}}$ engendre H. Soit $d < m$ tel que $d \mid m$. Étant donnée l'égalité

$$\frac{q^m - 1}{q^d - 1} = \frac{\prod_{k_1 \mid m} \Phi_{k_1}(q)}{\prod_{k_2 \mid d} \Phi_{k_2}(q)},$$

on voit facilement que $\Phi_m(q) \mid \frac{q^m-1}{q^d-1}$. Ainsi,

$$\mathrm{N}_{\mathrm{K}/\mathbf{F}_{q^d}}\left(g^{\frac{q^m-1}{\Phi_m(q)}}\right) = 1,$$

et $\mathrm{H} \subset \mathrm{T}_m(\mathbf{F}_q)$. Pour l'autre inclusion, on se sert du fait[1] suivant :

> *L'idéal de $\mathbf{Z}[\mathrm{X}]$ engendré par les polynômes $\frac{\mathrm{X}^m-1}{\mathrm{X}^{m/p}-1}$, où p parcourt l'ensemble des diviseurs premiers de m est l'idéal principal engendré par $\Phi_m(\mathrm{X})$.*

(1). Ce lemme a été montré par De Bruijn dans [dB53] et une démonstration plus récente se trouve dans [dS00].

Ainsi, $\phi_m(q)$ peut être exprimé comme combinaison linéaire de termes de la forme $\frac{q^m-1}{q^d-1}$ avec $d \mid m$. Comme h est un élément de $T_m(\mathbf{F}_q)$, on a bien $h^{\phi_m(q)} = 1$.

Pour la deuxième assertion, soit $h \in T_m(\mathbf{F}_q)$ un élément d'ordre ne divisant pas m. Supposons que $h \in \mathbf{F}_{q^d}$ avec $d < m$ et $d \mid m$, alors

$$N_{K/\mathbf{F}_{q^m}}(h) = h^{m/d}.$$

Comme d'autre part, $N_{K/\mathbf{F}_{q^d}}(h) = 1$, l'ordre de h divise m/d donc m, et on a une contradiction. □

Le groupe $T_m(\mathbf{F}_q)$ introduit par la définition III – 10 peut aussi être vu comme l'ensemble des points $\mathbf{F}_q$–rationnels d'un tore algébrique, T_m, défini comme variété algébrique affine (cf. [RS03], "Definition 6"). La dimension de T_m vu comme variété est alors $\varphi(m)$, en notant φ l'indicatrice d'Euler. Dans la suite, on appellera tore algébrique le groupe $T_m(\mathbf{F}_q)$ ainsi que la variété T_m.

En voyant $T_m(\mathbf{F}_q)$ comme un sous-groupe de F_{q^m}, on peut représenter ses éléments par m éléments de F_q. Grâce à l'interprétation en terme de variété, si l'on a une paramétrisation rationnelle de la variété T_m sur F_q, on aura besoin de seulement $\varphi(m)$ points. Précisons la notion de paramétrisation rationnelle.

Rationalité du tore algébrique sur $\mathbf{F}_q$

Définition III – 12. *Soit $\mathbf{A}^{\varphi(m)}$ l'espace affine de dimension $\varphi(m)$. Le tore algébrique T_m sur $\mathbf{F}_q$ est dit rationnel s'il existe des sous ensembles ouverts pour la topologie de Zariski, $W \subset T_m$ et $U \subset \mathbf{A}^{\varphi(m)}$ et des fonctions rationnelles $\rho_1, \ldots, \rho_{\varphi(m)} \in \mathbf{F}_q(x_1, \ldots, x_m)$ et $\psi_1, \ldots, \psi_m \in \mathbf{F}_q(y_1, \ldots, y_{\varphi(m)})$ tels que $\rho = (\rho_1, \ldots, \rho_{\varphi(m)}) : W \to U$ et $\psi = (\psi_1, \ldots, \psi_m) : U \to W$ soient des isomorphismes réciproques. La fonction ρ est alors appelée une paramétrisation rationnelle de T_m.*

Fait : *Le tore T_m est rationnel si m est une puissance d'un nombre premier ou le produit de deux puissances de nombres premiers. Il est conjecturé rationnel pour tout entier m.*

Si le tore T_m est rationnel, l'ouvert W étant au moins de dimension 1, $T_m \setminus W$ est au plus de dimension $\varphi(m) - 1$. Ainsi, la proportion de points non couverts par la paramétrisation rationnelle est de l'ordre de $1/q$ donc négligeable dès que q a des tailles cryptographiques.

Pour les utilisations cryptographiques, on utilise dans le groupe $T_m(\mathbf{F}_q)$ les cryptosystèmes basés sur le logarithme discret. On utilise alors les fonctions ρ et ψ comme fonctions de compression/décompression : les chiffrés et les clefs sont stockés au moyen de $\varphi(m)$ éléments de $\mathbf{F}_q$ et décompressés au moyen de la fonction ψ en des éléments de $T_n(\mathbf{F}_{q^m}) \subset \mathbf{F}_{q^m}$. Les calculs sont ensuite faits dans F_{q^m} puis les résultats re-compressés grâce à la fonction ρ (l'idéal serait de pouvoir calculer directement avec les éléments compressés, on verra que cela est possible dans le cas $m = 2$). La rationalité de T_m doit donc être explicite.

L'efficacité de la compression est de $\varphi(m)/m$. Étudions comment elle varie suivant les valeurs de m. Si m se décompose en produits de nombres premiers p_i distincts :

$$m = \prod_{i \in \mathrm{I}} p_i^{\alpha_i},$$

où les α_i, avec i élément de I, sont tous des entiers naturels non nuls, on a

$$\frac{\varphi(m)}{m} = \prod_{i \in \mathrm{I}} \frac{p_i - 1}{p_i}.$$

La compression ne dépend donc que des premiers p_i et ne dépend pas des valuations α_i et sera d'autant plus efficace que les premiers sont nombreux et petits. En effet, la fonction $x \mapsto \frac{x-1}{x}$ étant croissante et majorée par 1 sur $\mathbf{R}^{\times +}$, on a pour deux premiers distincts :

$$\frac{p_1 - 1}{p_1} < \frac{p_2 - 1}{p_2}, \text{ si } p_1 < p_2,$$

et, pour $k+1$ premiers distincts,

$$\prod_{i=1}^{k+1} \frac{p_i - 1}{p_i} < \prod_{i=1}^{k} \frac{p_i - 1}{p_i}.$$

On donne les valeurs intéressantes (correspondant au produit des premiers nombres premiers) dans le tableau suivant :

m	2	6	30	210	2310
$\varphi(m)/m$	0.5	≈ 0.33	≈ 0.27	≈ 0.23	≈ 0.21

Notons qu'en pratique seuls les cas $m = 2$ et $m = 6$ sont utilisables puisque au delà la rationalité n'est pas connue. Dans [vDGP+05] on donne cependant une manière de contourner cette restriction en suivant une idée de [vDW04] (pseudo-rationalité).

Le tore $\mathrm{T}_2(\mathbf{F}_q)$

On considère le corps fini $\mathbf{F}_q$ de caractéristique différente de 2 et une extension de degré deux, $\mathbf{F}_{q^2}$. On note Δ un entier non carré tel que $\mathbf{F}_{q^2} = \mathbf{F}_q(\sqrt{\Delta})$ et σ l'élément non trivial du groupe de Galois de l'extension $\mathbf{F}_{q^2}/\mathbf{F}_q$. On notera aussi $\overline{\alpha} = \sigma(\alpha)$ le conjugué d'un élément α de $\mathbf{F}_{q^2}$.

Par définition, le tore $\mathrm{T}_2(\mathbf{F}_q)$ est le sous-groupe de $\mathbf{F}_{q^2}^{\times}$ constitué des éléments de norme 1 sur $\mathbf{F}_q$. Dans le cas où $q = p$, avec p premier, on retrouve le groupe $(\mathcal{O}_\Delta / p\mathcal{O}_\Delta)^\wedge$ dans le cas particulier $\left(\frac{\Delta}{p}\right) = -1$.

La variante du cryptosystème LUC (cf. [SS94]), basée sur le problème du logarithme discret, ne manipule que des traces d'éléments de $(\mathcal{O}_\Delta / p\mathcal{O}_\Delta)^\wedge$ sur $\mathbf{F}_p$. Pour cela, elle utilise

seulement la suite de Lucas V définie en sous-section 3.2. Soit $g \in (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\wedge$, si l'on connaît $\mathrm{Tr}(g)$, on connaît alors g et $\sigma(g)$ qui sont racines du polynôme de $\mathrm{F}_p[\mathrm{X}]$, $\mathrm{X}^2 - \mathrm{Tr}(g)+1$. Ainsi, ce cryptosystème utilise les éléments du tore $\mathrm{T}_2(\mathbf{F}_p)$ modulo l'action de σ et la trace permet de compresser ces éléments en un seul élément du corps de base, d'où un rapport de compression de $1/2$.

Voyons comment utiliser la rationalité du tore $\mathrm{T}_2(\mathbf{F}_q)$ pour travailler directement avec ses éléments en ayant toujours une compression de $1/2$. On reprend le cas général avec q puissance d'un nombre premier. Soit ψ l'application :

$$\begin{array}{rccc} \Psi : & \mathbf{F}_{q^2}^\times & \longrightarrow & \mathrm{T}_2(\mathbf{F}_q) \\ & \alpha & \longmapsto & \alpha/\overline{\alpha} \end{array}$$

Cette application Ψ est bien définie et c'est un morphisme de groupes multiplicatifs. Par le théorème 90 de Hilbert, Ψ est surjective. D'autre part, $\ker\Psi = \mathbf{F}_q^\times$. Pour engendrer $\mathrm{T}_2(\mathbf{F}_q)$, on peut donc se restreindre à des éléments de $\mathbf{F}_{q^2}^\times$ de classes différentes modulo $\mathbf{F}_q^\times$. Cela peut se faire en considérant le système de représentants :

$$\{1\} \cup \left\{a + \sqrt{\Delta},\, a \in \mathbf{F}_q\right\}.$$

On a construit une application surjective ψ de $\mathbf{A}^1(\mathbf{F}_q)$ dans $\mathrm{T}_2(\mathbf{F}_q)\setminus\{1\}$:

$$\psi : a \longmapsto \frac{a+\sqrt{\Delta}}{a-\sqrt{\Delta}}.$$

Pour l'application réciproque, on remarque que si β est un élément de $\mathrm{T}_2(\mathbf{F}_q)$, on a $1+\beta = \beta(\overline{\beta}+1)$. Donc, si $\beta \neq -1$, on a

$$\beta = \frac{1+\beta}{1+\overline{\beta}},$$

et en notant $\beta = \beta_1 + \beta_2\sqrt{\Delta}$, on obtient

$$\beta = \psi\left(\frac{1+\beta_1}{\beta_2}\right),$$

si $\beta_2 \neq 0$, *i.e.*, si $\beta \neq \pm 1$. On définit donc ρ de $\mathrm{T}_2(\mathbf{F}_q)\setminus\{\pm1\}$ dans $\mathbf{A}^1(\mathbf{F}_q)\setminus\{0\}$:

$$\rho : \beta = \beta_1 + \beta_2\sqrt{\Delta} \longmapsto \frac{1+\beta_1}{\beta_2}.$$

Comme ρ et ϕ sont des isomorphismes réciproques de $\mathrm{T}_2(\mathbf{F}_q)\setminus\{\pm1\}$ dans $\mathbf{A}^1(\mathbf{F}_q)\setminus\{0\}$, on a trouvé une paramétrisation rationnelle de $\mathrm{T}_2(\mathbf{F}_q)$ au sens de la définition III – 12. Notons que l'on peut étendre cet isomorphisme entre $\mathrm{T}_2(\mathbf{F}_q)$ et $\mathbf{P}^1(\mathbf{F}_q)$ en envoyant 1 sur ∞ et -1 sur 0.

En plus d'utiliser ces fonctions à fin de compression/décompression, on peut transporter la loi de groupe de $\mathrm{T}_2(\mathrm{F}_q)$ dans $\mathbf{P}^1(\mathbf{F}_q)$. Soient a et b deux éléments de $\mathbf{P}^1(\mathbf{F}_q)$, on définit une loi de groupe $\star$ dans $\mathbf{P}^1(\mathbf{F}_q)$ en posant

$$a \star b := \rho\big(\psi(a)\,\psi(b)\big).$$

Ici, un calcul rapide donne que $\psi(a)\psi(b) = \psi\left(\frac{ab+\Delta}{a+b}\right)$ pour a et b dans $\mathbf{F}_q^\times$ tels que $a \neq -b$. On a donc,

$$\forall a, b \in \mathbf{F}_q^\times,\ a \neq -b,\ a \star b = \frac{ab+\Delta}{a+b}.$$

L'avantage de la loi $\star$ est de pouvoir utiliser des cryptosystèmes dans $\mathrm{T}_2(\mathrm{F}_q)$ en manipulant des éléments de tailles deux fois moindre qu'en travaillant directement dans $\mathbf{F}_{q^2}$. Cependant le calcul de $a \star b$ prend deux multiplications et une inversion dans $\mathbf{F}_q$ alors que les multiplications dans $\mathbf{F}_{q^2}$ prennent 5 multiplications dans $\mathbf{F}_q$ et sont donc plus rapides. Concernant les exponentiations, le calcul de $a^{\star k}$ par "double and add" prend $\frac{3}{2}|k|_2$ inversions et $2|k|_2$ multiplications dans $\mathbf{F}_q$. Dans le cas $q = p$, avec p premier, par les suites de Lucas, le calcul de α^k dans $\mathbf{F}_{q^2}$ avec $\alpha \in \mathrm{T}_2(\mathbf{F}_q)$ prend $3|k|_2$ multiplications et est donc plus efficace.

Tout ceci concerne des cryptosystèmes travaillant dans $\mathrm{T}_2(\mathbf{F}_q)$, si on travaille avec les traces d'éléments comme dans LUC, on obtient une exponentiation encore plus rapide. Par contre on perd la possibilité de multiplier des éléments et on ne peut pas faire des systèmes homomorphiques.

Si $n = pq$ est un entier RSA on peut travailler dans $\mathrm{T}_2(\mathbf{F}_p) \times \mathrm{T}_2(\mathrm{F}_q)$ si

$$\left(\frac{\Delta}{p}\right) = \left(\frac{\Delta}{q}\right) = -1,$$

en utilisant la loi $\star$ modulo n, tout comme on travaillait dans $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$, mais les calculs sont toujours moins efficaces à causes des inversions.

3.5. Générateur de $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\wedge$

Dans cette sous-section, on note Δ un entier non carré de $\mathbf{Z}$ et p un premier impair ne divisant pas Δ. On donne un algorithme de recherche d'un générateur de $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\wedge$.

On étudie tout d'abord l'application suivante qui permet de créer des éléments de norme 1 :

$$\begin{array}{rccc} \Psi : & (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\times & \longrightarrow & (\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\wedge \\ & \alpha = a + b\sqrt{\Delta} & \longmapsto & \frac{a+b\sqrt{\Delta}}{a-b\sqrt{\Delta}} \end{array}$$

Cette application Ψ généralise celle vue pour le tore algébrique (*i.e.*, dans le cas où Δ n'est pas un carré modulo p), page 65. Comme précédemment, Ψ est bien définie et c'est un morphisme surjectif. Son noyau est toujours $(\mathbf{Z}/p\mathbf{Z})^\times$ et un système de représentants de $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\times/(\ker\Psi)$ sera

$$\{1\} \cup \left\{a+\sqrt{\Delta}, a \in (\mathbf{Z}/p\mathbf{Z}) \setminus \{\pm b\}, b^2 = \Delta\right\},$$

la restriction $a \neq \pm b$ n'intervenant que si $\left(\frac{\Delta}{p}\right) = 1$. Le calcul de $\Psi(\alpha)$ avec $\alpha = a+\sqrt{\Delta}$ se fait en deux multiplications, un carré et une inversion dans $\mathbf{Z}/p\mathbf{Z}$:

$$\Psi(\alpha) = \frac{a^2+\Delta}{a^2-\Delta} + \frac{2a}{a^2-\Delta}\sqrt{\Delta}.$$

On cherche maintenant un élément g de ce système de représentants tel que $\Psi(g)$ engendre $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\wedge$. D'après ce qui précède, il faut que g soit d'ordre

$$\left(p - \left(\frac{\Delta}{p}\right)\right)$$

modulo $(\mathbf{Z}/p\mathbf{Z})^\times$. Notons k cet ordre. Si $(\Delta/p) = 1$, on a $g^{p-1} = 1$ et si $(\Delta/p) = -1$, $\mathrm{N}(g) = g^{p+1} \in (\mathbf{Z}/p\mathbf{Z})^\times$. Dans tous les cas, on a

$$k \,\Big|\, \left(p - \left(\frac{\Delta}{p}\right)\right),$$

et $\Psi(g)$ est un générateur si et seulement si $k = p - \left(\frac{\Delta}{p}\right)$.

Ainsi, on dispose d'un test pour savoir si un élément donné est un générateur du groupe $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\wedge$: un élément g ne sera pas un générateur s'il existe un entier k divisant strictement $\left(p-(\Delta/p)\right)$ tel que $g^k \in (\mathbf{Z}/p\mathbf{Z})^\times$. Si $g = a+\sqrt{\Delta}$, d'après la proposition III – 8, page 57, cela se traduit par $\mathrm{U}_k\left(2a, \mathrm{N}(g)\right) = 0$. On peut donc adapter un algorithme de recherche exhaustive de générateur de $(\mathbf{Z}/p\mathbf{Z})^\times$ (voir par exemple, [Coh93], 1.4.4, page 25) à notre cas. L'algorithme correspondant est donné dans la figure III.6, page suivante.

Remarque. Comme pour la recherche d'un générateur de $(\mathbf{Z}/p\mathbf{Z})^\times$, on peut simplifier l'algorithme en utilisant un p de forme spéciale, *i.e.*, tel que $\left(p-(\Delta/p)\right)/2$ soit premier.

Recherche d'un générateur $(\mathcal{O}_\Delta / p\mathcal{O}_\Delta)^\wedge$

entrée : p premier impair et Δ un entier non carré dans $\mathbf{Z}$ tel que $p \nmid \Delta$.

étape 0 : On cherche les facteurs premiers distincts $p_1, p_2, \ldots, p_l$ intervenant dans la factorisation de $p - \left(\frac{\Delta}{p}\right)$, $a := 0$,

étape 1 : $a := a + 1, i := 1$.

étape 2 : **si** $\left(\frac{\Delta}{p}\right) = 1$ *et* $a^2 \equiv \Delta \pmod{p}$ **alors** retourner à l'étape 1.

étape 3 : **si** $U_{\frac{p-(\Delta/p)}{p_i}}(2a, a^2 - \Delta) \bmod p = 0$ **alors** retourner à l'étape 1,
sinon $i := i + 1$.

étape 4 : **si** $i > l$ **alors** retourner $\Psi\left(a + \sqrt{\Delta}\right)$, **sinon** retourner à l'étape 2.

FIG. III.6 – Algorithme de recherche d'un générateur de $(\mathcal{O}_\Delta / p\mathcal{O}_\Delta)^\wedge$

CHAPITRE IV

CRYPTOSYSTÈMES

Dans ce chapitre, on va utiliser les trois familles de fonctions trappe définies au chapitre II dans les groupes finis étudiés au chapitre III. On va ici retrouver de nombreux cryptosystèmes proposés ces dernières années autour des idées de Paillier (cf. [Pai99] et page 84). Ce sera aussi l'occasion de montrer que les quotients de corps quadratiques permettent de construire des cryptosystèmes probabilistes relativement compétitifs.

Dans tout le chapitre, on notera n un entier RSA et s un entier naturel. On notera p et q les nombres premiers intervenant dans la factorisation de l'entier n.

Pour se mettre dans le cadre des hypothèses des fonctions trappe définies au chapitre II, on travaillera avec les groupes $(\mathbf{Z}/n^s\mathbf{Z})^\times$, E_{n^s} et $(\mathscr{O}_\Delta/n^s\mathscr{O}_\Delta)^\wedge$, où s est un entier strictement supérieur à 1. La sécurité de tous les systèmes étudiés sera basée sur la difficulté du problème de la factorisation de n.

Dans la première section, on va décrire des systèmes non probabilistes ce qui nous permettra de faire apparaître des fonctions trappe déterministes nécessaires pour utiliser les fonctions trappe probabilistes décrites en sous-sections 2.1 et 2.2 du chapitre II. Dans la deuxième section, on s'intéressera à des cryptosystèmes probabilistes homomorphiques, en utilisant la fonction trappe décrite en II–1.2. Enfin, dans la troisième section, on décrira des cryptosystèmes probabilistes non homomorphiques, en utilisant les deux familles de fonctions trappe définies en II–2.

1. Cryptosystèmes non probabilistes

Dans cette section, on va travailler modulo n. Les cryptosystèmes décrits seront issus de généralisations du système RSA. On va se servir du fait suivant.

Si $(\mathrm{G}, \times)$ est un groupe fini commutatif et si e est un entier premier avec un exposant μ du groupe G ($|\mathrm{G}|$ par exemple), alors le morphisme

$$f_e : \begin{cases} \mathrm{G} & \longrightarrow & \mathrm{G} \\ \rho & \longmapsto & \rho^e \end{cases}$$

est un isomorphisme d'inverse

$$f_d : \begin{cases} \mathrm{G} & \longrightarrow & \mathrm{G} \\ \rho & \longmapsto & \rho^d \end{cases}$$

où d est un entier vérifiant $d \equiv e^{-1} \pmod{\mu}$.

Dans $(\mathbf{Z}/n\mathbf{Z})^\times$, ces fonctions définissent le système RSA. Dans la suite, on va étudier le cas des courbes elliptiques et celui des quotients de corps quadratiques. Plus précisément, on va utiliser les fonctions f_e et f_d avec $\mathrm{G} = \mathrm{E}_n$ en sous-section 1.1 et avec $\mathrm{G} = (\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ en sous-section 1.2. Enfin, en sous-section 1.3, on comparera l'efficacité des cryptosystèmes obtenus.

1.1. Dans E_n, le système KMOV

On va donc utiliser la famille de fonctions f dans le groupe E_n. Ce groupe a fait l'objet d'une étude particulière en sous-section III–2.2. On reprend les notations de cette sous-section. On y a vu que le fait que la factorisation de n est inconnue a deux conséquences : on travaille en fait dans un sous-ensemble E'_n de E_n et, pour utiliser la fonction f, on doit construire les courbes en fonction du message à chiffrer. Tout ceci donne le système KMOV décrit dans [KMOV91].

Si les courbes varient suivant le message, il faut quand même maîtriser l'ordre du groupe des points de ces courbes afin de pouvoir bien choisir les paramètres e et d des fonctions f. Dans [KMOV91], les auteurs utilisent un choix particulier de nombres premiers p et q. On a en effet le lemme suivant, prouvé dans l'exercice 2.17, page 26 de [Men93].

Lemme IV – 1. *Si p est un nombre premier tel que $p \equiv 2 \pmod 3$, alors $\forall b \in (\mathbf{Z}/p\mathbf{Z})^\times$, $\mathrm{E}_p(0, b)$ est cyclique d'ordre $p+1$.*

Le module public du système est donc l'entier RSA, $n := pq$ avec p et q deux nombres premiers congrus à 2 modulo 3. On considère ensuite des courbes du type $\mathrm{E}_n(0, b)$. Un exposant de ce groupe est alors $\mu := \mathrm{ppcm}(p+1, q+1)$. L'exposant public e est choisi tel que $\mathrm{pgcd}(e, \mu) = 1$ et l'exposant secret d est l'inverse de e modulo μ.

Soit $\mathrm{M} := (\mathrm{M}_x, \mathrm{M}_y)$ un message à chiffrer où M_x et M_y sont des éléments de $\mathbf{Z}/n\mathbf{Z}$; on voit M comme un point de l'ensemble $\mathrm{E}'_n(0, b)$ inclus dans la courbe $\mathrm{E}_n(0, b)$ avec $b := \mathrm{M}_y^2 - \mathrm{M}_x^3$ (sous réserve que $\mathrm{pgcd}(n, b) = 1$). On a alors

$$\mathrm{C} := \mathrm{KMOV}_{n,e}(\mathrm{M}) := f_e(\mathrm{M}) = e.\mathrm{M}.$$

Les calculs sont faits dans $\mathrm{E}'_n(0, b)$ comme indiqué en sous-section III–2.2. Remarquons que b n'a même pas à être calculé car il n'intervient pas dans les formules classiques d'addition et de double dans une courbe elliptique. Cependant, on peut toujours retrouver b car $b = \mathrm{C}_y^2 - \mathrm{C}_x^3$. Pour déchiffrer, on calcule

$$d.\mathrm{C} = f_d(\mathrm{C}) = \big(f_d \circ f_e\big)(\mathrm{M}) = \mathrm{M},$$

toujours dans $\mathrm{E}'_n(0,b)$. Notons que ce déchiffrement peut être optimisé en utilisant le théorème des restes chinois, *i.e.*, en calculant $d.\mathrm{M}$ dans $\mathrm{E}_p(0,b)$ et dans $\mathrm{E}_q(0,b)$ pour récupérer M modulo p et q.

Ainsi, la fonction $\mathrm{KMOV}_{n,e}$ est une fonction trappe déterministe, permutation de l'ensemble

$$\left\{(x,y) \in \mathbf{Z}/n\mathbf{Z} \times \mathbf{Z}/n\mathbf{Z},\ \left(y^2 - x^3\right) \in (\mathbf{Z}/n\mathbf{Z})^\times\right\}.$$

Dans la section 3, on utilisera cette fonction en utilisant des y inversibles. On s'appuiera alors sur la proposition suivante.

Proposition IV – 2. *Soient $n := pq$ un entier RSA tel que $p \equiv q \equiv 2 \pmod 3$ et e un entier premier avec* $\mathrm{ppcm}(p+1, q+1)$. *La fonction* $\mathrm{KMOV}_{n,e}$ *est une permutation de l'ensemble*

$$\left\{(x,y) \in \mathbf{Z}/n\mathbf{Z} \times (\mathbf{Z}/n\mathbf{Z})^\times,\ \left(y^2 - x^3\right) \in (\mathbf{Z}/n\mathbf{Z})^\times\right\}.$$

Démonstration. Soit $\mathrm{M} := (\mathrm{M}_x, \mathrm{M}_y)$ un couple de $\left(\mathbf{Z}/n\mathbf{Z}\right)^2$ tel que $(y^2 - x^3) \in (\mathbf{Z}/n\mathbf{Z})^\times$ et tel que l'ordonnée y soit non inversible. On note $(\mathrm{C}_x, \mathrm{C}_y) := \mathrm{KMOV}_{n,e}(\mathrm{M})$. Il suffit de vérifier que C_y est non inversible. Modulo p, on doit vérifier que $(\mathrm{C}_x, \mathrm{C}_y) \equiv (\mathrm{C}_x, 0) \pmod p$ si $(\mathrm{M}_x, \mathrm{M}_y) \equiv (\mathrm{M}_x, 0) \pmod p$. Ceci signifie que modulo p, la fonction $\mathrm{KMOV}_{n,e}$ envoie un point d'ordre 2 de la courbe $\mathrm{E}_p(0, -\mathrm{M}_x^3)$ sur un point d'ordre 2 de cette même courbe. Cela est vrai car cette fonction conserve l'ordre des éléments de la courbe $\mathrm{E}_p(0, -\mathrm{M}_x^3)$ étant donné qu'elle correspond à un automorphisme de cette courbe. □

1.2. Dans $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$

Dans cette sous-section, Δ désigne un entier non carré premier avec n. On utilise la fonction f, définie page 69, dans le groupe $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. D'après le théorème III – 7 page 52, le groupe $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ est le produit de deux groupes cycliques $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\wedge$ et $(\mathcal{O}_\Delta/q\mathcal{O}_\Delta)^\wedge$, d'ordres respectifs $\varphi_\Delta(p) = p - (\Delta/p)$ et $\varphi_\Delta(q) = q - (\Delta/q)$. On note

$$\lambda_\Delta(n) := \mathrm{ppcm}\big(\varphi_\Delta(p), \varphi_\Delta(q)\big),$$

un exposant du groupe $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$.

Adaptation de RSA dans $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$

On se retrouve dans la même situation qu'avec le groupe E_n : si on ne connaît pas la factorisation de n, il est difficile de produire directement des éléments de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. En effet, il faut trouver x et y dans $\mathbf{Z}/n\mathbf{Z}$ tels que $x^2 - \Delta y^2 = 1$. On donne ici une méthode utilisant des groupes variant suivant le message à chiffrer. Une adaptation alternative utilisant des groupes fixes sera donnée en page 77.

Contrairement à ce que l'on a vu dans les courbes elliptiques, l'ordre des groupes variables ne posera pas de problème, on n'impose donc aucune restriction sur p et q.

Le module public est l'entier RSA n. Un message à chiffrer est de la forme (m_1, m_2), un couple d'entiers naturels strictement inférieurs à n tel que $(m_1^2 - 1)$ et m_2 soient premiers avec n. Ainsi, en choisissant Δ tel que

$$\Delta \equiv (m_1^2 - 1)m_2^{-2} \pmod{n},$$

on aura $\mathrm{pgcd}(\Delta, n) = 1$ et, modulo $n\mathscr{O}_\Delta$, $m_1 + m_2\sqrt{\Delta}$ sera bien un élément de $(\mathscr{O}_\Delta/n\mathscr{O}_\Delta)^\wedge$.

L'exposant public est un entier e premier avec $(p^2 - 1)(q^2 - 1)$. Ainsi, quelque soit le signe des symboles de Legendre (Δ/p) et (Δ/q), l'entier e sera premier avec $\lambda_\Delta(n)$. On pose

$$\Lambda := \Big\{(x, y) \in \mathbf{N} \times \mathbf{N},\, x < n,\, y < n,\, \mathrm{pgcd}(x^2 - 1, n) = \mathrm{pgcd}(y, n) = 1\Big\},$$

et on définit la fonction de chiffrement, appelée dans la suite fonction RSA quadratique :

$$\mathrm{RSA\text{–}Q}_{n,e} : \begin{cases} \Lambda & \longrightarrow & \Lambda \\ (m_1, m_2) & \longmapsto & (c_1, c_2) \end{cases}$$

où (c_1, c_2) est tel que

$$c_1 + c_2\sqrt{\Delta} \;\equiv\; \left(m_1 + m_2\sqrt{\Delta}\right)^e \pmod{n\mathscr{O}_\Delta},$$

l'exponentiation à la puissance e étant faite dans le groupe $(\mathscr{O}_\Delta/n\mathscr{O}_\Delta)^\wedge$, avec Δ un entier non carré tel que $\Delta \equiv (m_1^2 - 1)m_2^{-2} \pmod{n}$.

On a alors la proposition suivante.

Proposition IV – 3. *Soient $n := pq$ un entier RSA et e un entier premier avec (p^2-1, q^2-1). La fonction* $\mathrm{RSA\text{–}Q}_{n,e}$ *définie précédemment est une permutation de l'ensemble Λ.*

Démonstration. On montre d'abord que la fonction est bien définie. Pour cela, on note (c_1, c_2) l'image par $\mathrm{RSA\text{–}Q}_{n,e}$ d'un élément (m_1, m_2) de Λ. Supposons que c_2 ne soit pas premier avec n, car, par exemple, $c_2 \equiv 0 \pmod{p}$. Modulo $p\mathscr{O}_\Delta$, c_1 est alors un élément de $(\mathscr{O}_\Delta/p\mathscr{O}_\Delta)^\wedge$. On a donc $\mathrm{N}(c_1) \equiv 1 \pmod{p}$, *i.e.*, $c_1 \equiv \pm 1 \pmod{p}$. Ainsi, modulo $p\mathscr{O}_\Delta$, c_1 est d'ordre deux. Or, $\mathrm{RSA\text{–}Q}_{n,e}$ coïncide avec l'automorphisme f_e de $(\mathscr{O}_\Delta/p\mathscr{O}_\Delta)^\wedge$, donc conserve l'ordre des éléments. L'élément $m_1 + m_2\sqrt{\Delta}$ est donc d'ordre deux. Or,

$$\left(m_1 + m_2\sqrt{\Delta}\right)^2 = m_1^2 + m_2^2\Delta + 2m_1m_2\sqrt{\Delta}.$$

On a donc $m_1^2 + m_2^2\Delta \equiv 1 \pmod{p}$. Cependant, par le choix de Δ, on a aussi la congruence $m_1^2 + m_2^2\Delta \equiv 2m_1^2 - 1 \pmod{p}$. Comme p est impair, on a $m_1^2 \equiv 1 \pmod{p}$ ce qui contredit l'appartenance de (m_1, m_2) à Λ.

Ainsi, c_2 est bien premier avec n. Comme modulo $n\mathscr{O}_\Delta$, $c_1 + c_2\sqrt{\Delta}$ est un élément de $(\mathscr{O}_\Delta/n\mathscr{O}_\Delta)^\wedge$, on a $\Delta \equiv (c_1^2 - 1)/c_2^2 \pmod{n}$. Comme Δ et c_2 sont premiers avec n, $(c_1^2 - 1)$ aussi et $\mathrm{RSA\text{–}Q}_{n,e}$ est bien définie.

Pour finir, la fonction $\mathrm{RSA\text{–}Q}_{n,e}$ coïncidant avec un automorphisme de $(\mathscr{O}_\Delta/n\mathscr{O}_\Delta)^\wedge$, on voit immédiatement que c'est une permutation de Λ. □

On note (c_1, c_2) le chiffré d'un élément (m_1, m_2) de Λ. L'exposant privé est un entier d tel que d soit l'inverse de e modulo $(p^2-1)(q^2-1)$. En ayant choisi d ainsi on est sûr d'avoir la congruence $ed \equiv 1 \pmod{\lambda_\Delta(n)}$. Pour déchiffrer (c_1, c_2), on retrouve $\Delta \equiv (c_1^2-1)/c_2^2 \pmod{n}$, puis (m_1, m_2) en calculant la décomposition en base $(1, \sqrt{\Delta})$ de

$$\left(c_1 + c_2\sqrt{\Delta}\right)^d,$$

où le calcul se fait dans $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. On utilise ainsi la fonction RSA–$Q_{n,d}$.

On peut aussi retrouver la valeur de $\lambda_\Delta(n)$. En effet, modulo p par exemple, comme on a $\Delta \equiv (c_1^2-1)/c_2^2 \pmod{p}$, on a aussi

$$\left(\frac{\Delta}{p}\right) = \left(\frac{c_1^2-1}{p}\right).$$

Ceci permet d'obtenir un déchiffrement plus performant en utilisant les restes chinois et en choisissant les exposants privés dans le vecteur

$$\left(d_{(p,-1)}, d_{(p,1)}, d_{(q,-1)}, d_{(q,1)}\right),$$

où $d_{(p,i)} \equiv e^{-1} \pmod{p-i}$ pour $i = \pm 1$, et de même pour le nombre premier q.

Notons que les exponentiations peuvent se faire au moyen des suites de Lucas U et V (cf. proposition III – 8, page 57). On obtient un cryptosystème extrêmement proche de KMOV mais plus efficace grâce à cette utilisation des suites de Lucas, comme on le verra en sous-section 1.3.

Le système LUC

Le cryptosystème LUC a été proposé dans [SL93]. Le langage utilisé dans cet article (ainsi que dans les articles postérieurs étudiants LUC) est celui, un peu obscur, des suites de Lucas. Cependant, il est facile de voir que ce système est une amélioration immédiate de celui présenté précédemment.

Au lieu de calculer l'exponentiation complète $\alpha \mapsto \alpha^e$ dans $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$, on utilise seulement des traces d'éléments. À un entier m défini modulo n, on associe la trace de α^e où α est un élément de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ tel que $\mathrm{Tr}(\alpha) \equiv m \pmod{n}$. D'après la proposition III – 8 page 57, ce calcul peut se faire uniquement avec la suite de Lucas V. On a ainsi un gain calculatoire comparé au cryptosystème précédent où l'on doit utiliser les suites V et U.

La clef publique est toujours constituée d'un module RSA n et d'un entier naturel e premier avec $(p^2-1)(q^2-1)$. On pose

$$\Lambda := \left\{x \in \mathbf{N},\, x < n,\, \mathrm{pgcd}(x^2-4, n) = 1\right\},$$

et on définit la fonction de chiffrement LUC :

$$\mathrm{LUC}_{n,e} : \begin{cases} \Lambda & \longrightarrow & \Lambda \\ m & \longmapsto & \mathrm{V}_e(m) \bmod n \end{cases}$$

On a alors la propriété suivante :

Proposition IV – 4. *Soient $n := pq$ un entier RSA et e un entier naturel premier avec le produit $(p^2-1)(q^2-1)$. La fonction $\mathrm{LUC}_{n,e}$ est une permutation de Λ_n.*

Démonstration. On montre d'abord que $\mathrm{LUC}_{n,e}$ est bien définie. Soit m un élément de Λ et Δ un entier non carré tel que $\Delta \equiv x^2 - 4 \pmod{n}$. On a $\mathrm{pgcd}(\Delta, n) = 1$. Soit $\alpha \in \mathcal{O}_\Delta$ tel que

$$\alpha \equiv \frac{m + \sqrt{\Delta}}{2} \pmod{n\mathcal{O}_\Delta}.$$

Modulo n, α est un élément de norme 1 et vérifie $\mathrm{LUC}_{n,e}(m) \equiv \mathrm{V}_e(m) \equiv \mathrm{Tr}(\alpha^e) \pmod{n}$, d'après la proposition III – 8.

On note $(c_1, c_2) := \mathrm{RSA\text{–}Q}_{n,e}(m/2, 1/2)$. On a $\alpha^e \equiv c_1 + c_2\sqrt{\Delta} \pmod{n\mathcal{O}_\Delta}$. On a donc $\mathrm{V}_e(m) \equiv 2c_1 \pmod{n}$ ce qui prouve que $\mathrm{V}_e(m)$ est bien un élément de Λ comme n est impair.

Pour voir que $\mathrm{LUC}_{n,e}$ est une bijection on va exhiber sa réciproque. Soit d l'inverse de e modulo $\varphi_\Delta(n)$, d'après la proposition III – 8, on a

$$\alpha^e \equiv \frac{\mathrm{V}_e(m) + \mathrm{U}_e(m)\sqrt{\Delta}}{2} \pmod{n\mathcal{O}_\Delta}$$

et

$$\alpha^{ed} = (\alpha^e)^d \equiv \frac{\mathrm{V}_d\big(\mathrm{V}_e(m)\big) + \mathrm{U}_e(m)\mathrm{U}_d\big(\mathrm{V}_e(m)\big)\sqrt{\Delta}}{2} \pmod{n\mathcal{O}_\Delta}.$$

Comme on a aussi $\alpha^{ed} \equiv \alpha \pmod{n\mathcal{O}_\Delta}$, on doit avoir $\mathrm{V}_d\big(\mathrm{V}_e(m)\big) \equiv m \pmod{n}$. On en déduit que $\mathrm{LUC}_{n,d} \circ \mathrm{LUC}_{n,e} = \mathrm{Id}_\Lambda$. Comme e et d jouent un rôle symétrique, on en déduit que $\mathrm{LUC}_{n,e}$ est bien une permutation de Λ. □

On a vu dans la preuve que le déchiffrement se fait grâce à la fonction $\mathrm{LUC}_{n,d}$ où d est un inverse de e modulo $\varphi_\Delta(n)$. La clef privée est donc la même que pour la fonction RSA quadratique.

Pour utiliser cette fonction dans la section 3, on aura besoin du corollaire suivant.

Corollaire IV – 5 (de la proposition IV – 4)**.** *Avec les notations de la proposition, la fonction $\mathrm{LUC}_{n,e}$ est une permutation de l'ensemble*

$$\left\{x \in \mathbf{N},\, x < n,\, \gcd(x^2 - 4, n) = \gcd(x, n) = 1\right\}.$$

Démonstration. Il suffit de montrer que $\mathrm{LUC}_{n,e}$ laisse stable $\{x, \gcd(x,n) \neq 1\}$. Modulo p, on doit montrer que $\mathrm{LUC}_{n,e}(0) \equiv 0 \pmod{p}$. Comme e est premier avec (p^2-1), e est impair et

$$\mathrm{V}_e(0) \equiv \mathrm{Tr}\Big(\big(\sqrt{\Delta}/2\big)^e\Big) \equiv 0 \pmod{p},$$

avec $\Delta \equiv -4 \pmod{p}$. □

Généralisation du système LUC

On a déjà signalé, en page 64, l'existence d'une variante de LUC ([SS94]), reprenant l'idée de ne travailler qu'avec des traces d'éléments. Cette variante est basée sur le problème du logarithme discret dans le groupe des éléments de norme 1 d'une extension de degré deux de $\mathbf{F}_p$ où p est premier.

Ces idées ont été généralisées à des extensions cubiques. C'est le système GH proposé par Gong et Harn dans [GH99, GHW01]. Voyons cela plus en détail. Soit $\mathbf{F}_q$ un corps fini et

$$\mathrm{F}(\mathrm{X}) := \mathrm{X}^3 - a\mathrm{X}^2 + b\mathrm{X} - 1,$$

un polynôme irréductible de $\mathbf{F}_q[\mathrm{X}]$. On note α une racine de F dans $\mathbf{F}_{q^3}$. Par construction, on a $\mathrm{N}(\alpha) = 1$. Dans l'esprit de LUC on définit une suite V par

$$\forall k \in \mathbf{N},\ \mathrm{V}_k(\alpha) := \mathrm{V}_k(a,b) := \mathrm{Tr}(\alpha^k).$$

Ceci permet de manipuler la suite V paramétrée par deux éléments de $\mathbf{F}_q$ au lieu de travailler avec les éléments de $\mathbf{F}_{q^3}$ qui sont eux représentés par 3 éléments de $\mathbf{F}_q$ en utilisant une base de $\mathbf{F}_{q^3}$ sur $\mathbf{F}_q$.

On a $\mathrm{V}_0(\alpha) = \mathrm{Tr}(1) = 3$, $\mathrm{V}_1(\alpha) = \mathrm{Tr}(\alpha) = a$, et, par utilisation des fonctions symétriques des racines, $\mathrm{V}_2(\alpha) = \mathrm{Tr}(\alpha^2) = a^2 - 2b$, puis

$$\forall k \geq 3,\ \mathrm{V}_k(\alpha) = a\mathrm{V}_{k-1}(\alpha) - b\mathrm{V}_{k-2}(\alpha) + \mathrm{V}_{k-3}(\alpha).$$

L'élément α est un élément du groupe des éléments de norme 1 de $\mathbf{F}_{q^3}$. D'après le lemme III – 11, ce groupe, $\mathrm{T}_3(\mathrm{F}_q)$, est cyclique d'ordre $\phi_3(q) = q^2 + q + 1$. On aura donc $\mathrm{V}_{q^2+q+1}(\alpha) = \mathrm{V}_0(\alpha) = 3$.

Comme pour LUC, on a une « propriété de composition ». On voit facilement que $b = \mathrm{Tr}(\alpha^{-1})$ et que α^{-1} est racine du polynôme $\mathrm{X}^3 - b\mathrm{X}^2 + a\mathrm{X} - 1$ irréductible sur $\mathbf{F}_q$. La suite $\big(\mathrm{V}_k(b,a)\big)_{k\in\mathbf{N}}$ est associée à ce polynôme et pour tout entier positif e, $\mathrm{V}_e(b,a) = \mathrm{Tr}(\alpha^{-e})$. L'élément α^e est alors racine du polynôme

$$\mathrm{X}^3 - \mathrm{V}_e(a,b)\mathrm{X}^2 + \mathrm{V}_e(b,a)\mathrm{X} + 1.$$

Par conséquent, si d est un entier,

$$\mathrm{V}_d\big(\mathrm{V}_e(a,b), \mathrm{V}_e(b,a)\big) = \mathrm{Tr}\big(\alpha^{ed}\big) = \mathrm{V}_{ed}(a,b).$$

Grâce à ceci, Ghong et Harn adaptent dans [GH99] le système de Diffie-Hellman. Le schéma RSA est aussi adapté, dans l'esprit du cryptosystème LUC, mais le système résultant de cette adaptation semble peu praticable.

Il est à noter que le système XTR (cf. [LV00]) est un cas particulier de ce qui précède. Dans XTR, on utilise $\mathbf{F}_q = \mathbf{F}_{p^2}$ avec p premier. Un élément g du sous-groupe d'ordre $p^2 - p + 1$ de $\mathrm{T}_3(\mathbf{F}_{p^2})$ (d'ordre $p^4 + p^2 + 1$) est alors racine de

$$\mathrm{X}^3 - \mathrm{Tr}(g) + \mathrm{Tr}(g)^p - 1,$$

et pour tout entier n, g^n est racine de

$$\mathrm{X}^3 - \mathrm{Tr}(g^n) + \mathrm{Tr}(g^n)^p - 1,$$

cf. “Lemma 2.2.1” et “Lemma 2.3.4” de [LV00].

Polynômes de Dickson et suites de Lucas

On peut trouver une définition des polynômes de Dickson dans [LMT93], chapitre 2 ou section 7.2, page 355 de [LN97]. Étant donné un anneau commutatif unitaire $\mathscr{R}$, et a un élément de $\mathscr{R}$, le polynôme de Dickson de $\mathscr{R}[\mathrm{X}]$ de degré k, noté $g_k(x,a)$, est défini par

$$g_k(\mathrm{X},a) := \sum_{i=0}^{\lfloor k/2 \rfloor} \frac{k}{k-i}\binom{k-i}{i}(-a)^i \mathrm{X}^{k-2i}. \tag{IV.1}$$

Ces polynômes ont été utilisés dans le cas $\mathscr{R} = \mathbf{Z}/n\mathbf{Z}$ et $a = 1$ par Müller et Nöbauer pour construire un cryptosystème (cf. [MN81, MN86]). Cependant, à la seule différence que Müller et Nöbauer autorisent l'utilisation d'un module n non RSA, ce cryptosystème est le même que le cryptosystème LUC présenté précédemment. En effet, si P et Q sont deux entiers, on montre facilement (cf. "Lemma 2.3" de [LMT93]) que $g_1(\mathrm{P},\mathrm{Q}) = \mathrm{P}$, $g_2(\mathrm{P},\mathrm{Q}) = \mathrm{P}^2 - 2\mathrm{Q}$ et que pour tout $k \geqslant 2$,

$$g_{k+1}(\mathrm{P},\mathrm{Q}) = \mathrm{P}\,g_k(\mathrm{P},\mathrm{Q}) - \mathrm{Q}\,g_{k-1}(\mathrm{P},\mathrm{Q}).$$

Ainsi, le polynôme de Dickson coïncide avec la suite de Lucas V, *i.e.*, pour tout entier $k \geqslant 1$, pour tout couple d'entiers (P,Q) tel que $\mathrm{P}^2 - 4\mathrm{Q}$ ne soit pas carré dans $\mathbf{Z}$,

$$\mathrm{V}_k(\mathrm{P},\mathrm{Q}) = g_k(\mathrm{P},\mathrm{Q}).$$

Fonctions de Rédei et T_2

Les fonctions de Rédei ont également été proposées pour construire des cryptosystèmes (cf. [Var88, LM84]). On peut trouver une étude de ces fonctions dans [LMT93], section 2.5.

Elles sont définies comme suit. Soit p un nombre premier impair. Soit un entier Δ non carré de $\mathbf{Z}$ tel que Δ ne soit pas un carré modulo p. On définit deux polynômes de $\mathbf{Z}[\mathrm{X}]$, g_e et h_e tels que

$$\left(\mathrm{X} + \sqrt{\Delta}\right)^e = g_e(\mathrm{X}) + h_e(\mathrm{X})\sqrt{\Delta}.$$

La fonction rationnelle de Rédei est alors $f_e(\mathrm{X}) := g_e(x)/h_e(\mathrm{X})$. On montre qu'elle vérifie l'égalité

$$\left(\frac{\mathrm{X}+\sqrt{\Delta}}{\mathrm{X}-\sqrt{\Delta}}\right)^e = \frac{f_e(\mathrm{X})+\sqrt{\Delta}}{f_e(\mathrm{X})-\sqrt{\Delta}}.$$

De plus, cette fonction induit une permutation de $\mathbf{Z}/p\mathbf{Z} \cup \{\infty\}$ si e est premier avec $(p+1)$. La réciproque de f_e est la fonction f_d avec $ed \equiv 1 \pmod{(p+1)}$. Elle peut donc être utilisée pour un cryptosystème de type RSA en introduisant un autre nombre premier q, distinct de p.

On fait maintenant le lien entre cette fonction et la paramétrisation du tore T_2, vue page 64. Avec les notations de cette page, on a pour tout $m \in \mathbf{Z}/p\mathbf{Z}$,

$$\psi(m)^e = \left(\frac{m+\sqrt{\Delta}}{m-\sqrt{\Delta}}\right)^e = \frac{f_e(m)+\sqrt{\Delta}}{f_e(m)-\sqrt{\Delta}} = \psi\big(f_e(m)\big),$$

et

$$m^{\star e} = \rho\big(\psi(m)^e\big) = f_e(m).$$

La fonction f_e coïncide donc avec l'exponentiation de la loi de groupe du tore transportée sur $\mathbf{P}_1(\mathbf{Z}/p\mathbf{Z})$. On retrouve donc le fait que c'est une permutation de $\mathbf{P}_1(\mathbf{Z}/p\mathbf{Z})$ et l'expression de sa réciproque. Cependant, les algorithmes proposés pour calculer f_e (cf. [Mor95]) reviennent à calculer l'exponentiation par la loi $\star$ avec l'expression vue en page 66. Comme on l'avait remarqué, si on utilise la loi $\star$ modulo un entier RSA, on obtiendra un système moins efficace qu'en calculant directement dans T_2 la fonction $\alpha \mapsto \alpha^e$ avec les suites de Lucas.

On obtient donc un système similaire au système RSA quadratique, avec un surcoût calculatoire. Cependant, contrairement à ce dernier système, on travaille avec des groupes fixes (*i.e.*, Δ est fixé) et on manipule seulement un élément de $\mathbf{Z}/n\mathbf{Z}$ au lieu de deux. Notons que ces avantages peuvent être transférés au système RSA quadratique en utilisant les fonctions ρ et ψ seulement comme fonctions de compression/décompression et en faisant le calcul des exponentiations avec les suites de Lucas. Cette démarche peut être intéressante pour construire des systèmes probabilistes homomorphiques. On décrit brièvement cette solution dans le paragraphe suivant.

Variante de la fonction RSA quadratique avec un discriminant fixe

On utilise toujours un module RSA n. On rend également public un entier Δ tel qu'il soit premier avec n et qu'il ne soit ni un carré dans $\mathbf{Z}$, ni modulo p, ni modulo q. L'exposant public est un entier e premier avec $(p+1)(q+1)$.

Par le théorème des restes chinois et la paramétrisation du tore vue page 64, on a l'isomorphisme :

$$(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge \xrightarrow{\sim} \mathbf{P}_1(\mathbf{Z}/p\mathbf{Z}) \times \mathbf{P}_1(\mathbf{Z}/p\mathbf{Z}).$$

Dans la pratique, comme la factorisation de n n'est pas publique, on va travailler dans $(\mathbf{Z}/n\mathbf{Z})^\times$ vu comme sous-ensemble de $\mathbf{P}_1(\mathbf{Z}/p\mathbf{Z}) \times \mathbf{P}_1(\mathbf{Z}/p\mathbf{Z})$. Ainsi, d'une part on exclut tous les points faisant intervenir l'infini. Par la paramétrisation, ce sont les points correspondant aux éléments α de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ qui sont congrus à 1 modulo p ou à 1 modulo q. D'autre part, en ne prenant que des éléments inversibles, on exclut, toujours via la paramétrisation, les points correspondant aux éléments α de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ qui sont congrus à -1 modulo p ou à -1 modulo q. Ainsi, on ne garde que les éléments de la forme $\alpha_1 + \alpha_2\sqrt{\Delta}$ avec $\alpha_2 \in (\mathbf{Z}/n\mathbf{Z})^\times$.

La fonction de chiffrement devient

$$\text{RSA–Q}_{\Delta,n,e} : \begin{cases} (\mathbf{Z}/n\mathbf{Z})^\times & \longrightarrow & (\mathbf{Z}/n\mathbf{Z})^\times \\ m & \longmapsto & \rho\big(\psi(m)^e\big) \end{cases}$$

où les fonctions ψ et ρ sont similaires à celles utilisées page 64 : ψ est une fonction de $(\mathbf{Z}/n\mathbf{Z})^\times$ dans $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ qui, à un élément a, associe $\big(a+\sqrt{\Delta}\big)/\big(a-\sqrt{\Delta}\big)$. La fonction ρ est la réciproque de ψ : c'est une fonction de $\psi\big((\mathbf{Z}/n\mathbf{Z})^\times\big) \subset (\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ dans $(\mathbf{Z}/n\mathbf{Z})^\times$ qui, à $\beta = \beta_1 + \beta_2\sqrt{\Delta}$, associe $(1+\beta_1)/\beta_2$. Notons que le calcul $\psi(m)^e$ est effectué dans $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$.

Cette fonction est bien définie, car $\psi(m)^e$ est un élément de $\psi((\mathbf{Z}/n\mathbf{Z})^\times)$ (modulo p, par exemple, l'ordre de $\psi(m)$ ne divise pas 2, donc celui de $\psi(m)^e$ non plus). De plus, on vérifie facilement que c'est une bijection.

Le déchiffrement se fait avec la fonction RSA–$Q_{\Delta,n,d}$ où d est un inverse de e modulo $(p+1)(q+1)$.

On obtient donc un système similaire au système basé sur la fonction RSA–$Q_{n,e}$. Les différences sont l'utilisation d'un discriminant fixe et la taille des messages et des chiffrés (ce sont maintenant des éléments de $(\mathbf{Z}/n\mathbf{Z})^\times$). Coté efficacité, on rajoute le coût de l'évaluation des fonctions ψ et ρ : il faut compter trois multiplications et une inversion modulo n pour la fonction ψ et une inversion modulo n pour la fonction ρ.

1.3. Comparaison des systèmes

Sécurité

Les systèmes LUC et RSA quadratique présentés en sous-section 1.2 ont une sécurité équivalente. En effet, il est évident que si l'on sait inverser la fonction RSA quadratique, on sait inverser la fonction LUC. Pour la réciproque on utilise la proposition III – 8. De même, si l'on sait inverser la fonction RSA quadratique, on sait inverser la variante utilisant un discriminant fixe.

On sait peu sur la comparaison de la sécurité entre les systèmes LUC, KMOV et RSA. Tous ces systèmes peuvent être inversés globalement en connaissant l'inverse de l'exposant public e modulo l'ordre du groupe considéré. Dans tous les cas, la connaissance de cet inverse est équivalente à celle de la factorisation de n. Dans [BBL95], les auteurs montrent que si l'on sait résoudre une instance particulière du problème LUC, alors on peut résoudre le problème RSA. Plus précisément, si on cherche à déchiffrer $c \in (\mathbf{Z}/n\mathbf{Z})^\times$ par RSA, il suffit de déchiffrer $c + c^{-1}$ par LUC. Cependant, comme mentionné par les auteurs, ceci n'implique pas que le problème LUC est plus fort que RSA.

Certaines attaques sur RSA exploitant la structure polynomiale ou le caractère homomorphique de la fonction ont été adaptées à LUC et KMOV (cf. [JQ98]). Les problèmes respectifs d'inversion ponctuelle de ces trois fonctions peuvent être interprétés comme le problème de recherche d'une racine d'un polynôme défini sur $\mathbf{Z}/n\mathbf{Z}$.

Pour RSA, on doit trouver une racine du polynôme $X^e - c$. Pour LUC, le polynôme $V_e(X) - c$, également de degré e, s'obtient par l'expression (IV.1) d'un polynôme de Dickson. Pour KMOV, un résultat présent dans [Sil86] (exercice 3.7, page 105), permet d'exprimer l'abscisse de eP en fonction de celle de P par un polynôme de degré e^2.

Un résultat de Coppersmith (cf. [Cop96]) permet de retrouver les racines inférieures à $n^{1/d}$ d'un polynôme de degré d sur $\mathbf{Z}/n\mathbf{Z}$ en temps polynomial. Ce résultat semble indiquer que la complexité du problème de recherche de racines d'un polynôme P défini sur $\mathbf{Z}/n\mathbf{Z}$ est reliée au degré de P. Sous cet angle, LUC et RSA ont même niveau de sécurité avec un paramètre e, de taille $|e|_2$, que KMOV avec un paramètre de taille $\frac{|e|_2}{2}$.

Efficacité

On compare maintenant l'efficacité des cryptosystèmes. On note M le coût d'une multiplication modulo n. On estime qu'une multiplication modulo p coûte M/3, qu'une inversion modulo n coûte 10M et qu'un inversion modulo p coûte 10/3M. On désigne par L le temps de calcul d'un symbole de Legendre dans $\mathbf{Z}/p\mathbf{Z}$ avec $|p|_2 = |n|_2/2$. Ce calcul est de complexité $\mathcal{O}(|p|_2^2)$.

Pour LUC et la fonction RSA quadratique, on utilise les algorithmes respectivement donnés dans les figures III.3 et III.4, pages 55 et 56. Pour RSA on utilise l'algorithme classique "square and multiply". Par cet algorithme, le calcul de x^k dans $\mathbf{Z}/n\mathbf{Z}$ coûte en moyenne $\frac{3}{2}|k|_2$ M. Pour KMOV, on utilise les formules classiques d'addition et de double en coordonnées affines, avec le même algorithme. Ainsi, le coût du calcul de k.M est en moyenne de $\frac{11}{2}|e|_2\,\mathrm{M} + \frac{3}{2}|e|_2\,\mathrm{I} \approx 20\,|e|_2\,\mathrm{M}$.

On utilise systématiquement les restes chinois pour le déchiffrement. On résume les coûts obtenus dans le tableau suivant.

Cryptosystème	RSA	KMOV	RSA quadratique	LUC
Cadre	$m \in (\mathbf{Z}/n\mathbf{Z})^\times$	$\mathrm{M} \in \mathrm{E}_n$	$m \in (\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$	$m \in \mathrm{Tr}\big((\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge\big)$
Taille de l'entrée	$\lvert n\rvert_2$	$2\lvert n\rvert_2$	$2\lvert n\rvert_2$	$\lvert n\rvert_2$
Taille de la sortie	$\lvert n\rvert_2$	$2\lvert n\rvert_2$	$2\lvert n\rvert_2$	$\lvert n\rvert_2$
Expansion	1			
Chiffrement	$\frac{3}{2}\lvert e\rvert_2\,\mathrm{M}$	$20\lvert e\rvert_2\,\mathrm{M}$	$3\lvert e\rvert_2\,\mathrm{M}$	$2\lvert e\rvert_2\,\mathrm{M}$
Déchiffrement	$\left(\frac{\lvert n\rvert_2}{2}+1\right)\mathrm{M}$	$(7\lvert n\rvert_2+2)\mathrm{M}$	$(\lvert n\rvert_2+1)\mathrm{M}+2\mathrm{L}$	$\left(\frac{2\lvert n\rvert_2}{3}+1\right)\mathrm{M}+2\mathrm{L}$

On voit que le système LUC est de complexité très proche de RSA (on augmente d'un facteur 4/3), grâce à l'efficacité du calcul des termes de la suite de Lucas V. Pour RSA quadratique, comme on doit calculer aussi des termes de la suite U la complexité augmente encore d'un facteur 3/2. Le système KMOV est sept fois plus lent que le système RSA quadratique (on peut diviser ce coût par deux si on emploie un exposant e de taille deux fois plus petite comme évoqué dans le paragraphe précédent).

Ainsi, pour les systèmes basés sur la factorisation, les quotients de corps quadratiques semblent offrir une bien meilleure alternative à l'utilisation des quotients de $\mathbf{Z}$ que les groupes de points de courbes elliptiques.

2. Cryptosystèmes probabilistes homomorphiques

Dans cette section, on va utiliser la fonction trappe définie en sous-section II–1.2 pour produire des systèmes probabilistes homomorphiques. Dans la première sous-section, on va utiliser les quotients de $\mathbf{Z}$, ensuite on utilisera les courbes elliptiques et enfin les corps quadratiques.

Dans les deux premières sections, on décrira essentiellement des cryptosystèmes existants. Dans la troisième section, on énoncera un nouveau cryptosystème. Enfin, on résumera par un tableau récapitulatif les propriétés des principaux cryptosystèmes décrits.

On reprend les notations de la section II–1. Ainsi, G désignera un groupe abélien fini multiplicatif. Si k est un entier, G^k sera le sous-groupe de G constitué par les puissances k–ièmes.

2.1. Dans les quotients de Z

Dans cette sous-section, on va donc considérer les puissances k–ièmes de groupes G du type $(\mathbf{Z}/a\mathbf{Z})^\times$ en faisant varier les entiers k et a.

On va ainsi faire l'historique d'une certaine classe de cryptosystèmes, du système fondateur de la cryptographie probabiliste, le système de Goldwasser-Micali, au plus achevé de cette classe de système, le système de Paillier. En faisant cet historique, on rentrera progressivement dans le cadre des hypothèses de la section II–1.

Le système de Goldwasser-Micali (1984)

Dans [GM84], Goldwasser et Micali introduisent le premier système probabiliste. Ce cryptosystème ne rentre pas directement dans le cadre de la fonction trappe définie en section II–1, mais lui est cependant fortement lié.

Cadre : On pose $G = (\mathbf{Z}/n\mathbf{Z})^\times$ et $k = 2$. On considère ainsi le sous-groupe G^2 constitué des carrés. De cette manière, on ne vérifie pas les hypothèses de la section II–1. En effet, G est d'ordre $\varphi(n) = (p-1)(q-1)$ qui est divisible par 4. L'entier k n'est donc pas premier avec $|G|/k$. On ne va donc pas travailler avec G/G^2, mais avec un sous-groupe de ce groupe quotient.

On note H le sous-groupe de G constitué des éléments de symbole de Jacobi positif, *i.e.*, l'ensemble des éléments h tels que

$$\left(\frac{h}{n}\right) = \left(\frac{h}{p}\right) \times \left(\frac{h}{q}\right) = 1.$$

Ce groupe H contient donc les éléments de G^2 et les éléments qui ne sont des carrés ni modulo p, ni modulo q (les « faux carrés »).

Dans $(\mathbf{Z}/p\mathbf{Z})^\times$, on a $(p-1)/2$ carrés, donc, par l'isomorphisme du théorème des restes chinois, on a $\left|G^2\right| = \varphi(n)/4$ et $|H| = \varphi(n)/2$. Le quotient G/G^2 est donc d'ordre 4 isomorphe à $\mathbf{Z}/2\mathbf{Z} \times \mathbf{Z}/2\mathbf{Z}$ (tout élément de G/G^2 est d'ordre 2). Le quotient H/G^2 est alors un sous-groupe d'ordre 2 de G/G^2. C'est avec ce dernier groupe quotient que l'on va travailler.

Pour retrouver la situation de la section II–1, on a besoin d'avoir un générateur de ce groupe quotient. On note g un « faux carré » de G, *i.e.*, un élément de G qui n'est ni un carré modulo p, ni un carré modulo q. L'élément $\overline{g}$ de H/G^2 est alors un générateur.

Chiffrement : On utilise la fonction surjective suivante :

$$\mathscr{E}_{\mathbf{Z}/n\mathbf{Z},2,g} : \begin{cases} \{0,1\} \times (\mathbf{Z}/n\mathbf{Z})^\times & \longrightarrow & \mathrm{H} \\ (m \quad , \quad r) & \longmapsto & g^m r^2 \end{cases}$$

où m est le message à chiffrer et r est choisi au hasard.

Déchiffrement et sécurité : On note $c := \mathscr{E}_{\mathbf{Z}/n\mathbf{Z},2,g}(m,r)$ un chiffré d'un message m. Pour déchiffrer, il faut retrouver le logarithme discret de c en base $\overline{g}$ dans H/G^2. Ainsi, avec les notations de la section II–1, on a

$$m \equiv [\![c]\!]_g \pmod 2.$$

Autrement dit, on doit déterminer si c est un carré ou un faux carré. La sécurité du système repose donc sur le problème de résidualité quadratique. D'après le théorème II – 11, la sécurité sémantique repose sur le problème de reconnaissance des carrés dans H, c'est à dire, comme pour la sécurité, sur le problème de la résidualité quadratique.

La factorisation de n fournit une trappe pour déchiffrer. En effet, on a pour tout $a \in \{0,1\}$,

$$[\![c]\!]_g \equiv a \pmod 2 \iff \left(\frac{c}{p}\right) = (-1)^a.$$

Efficacité : Le chiffrement requiert au plus une multiplication et une élévation au carré dans $\mathbf{Z}/n\mathbf{Z}$, le déchiffrement requiert de calculer un symbole de Legendre. Ceci se fait en $\mathscr{O}(|p|_2^2)$.

Notons que la fonction $\mathscr{E}$ n'est pas un morphisme de groupes car g n'est pas forcément d'ordre 2 dans $(\mathbf{Z}/n\mathbf{Z})^\times$. Par contre, comme dans la section II–1, la fonction de déchiffrement s'exprime comme un logarithme discret. Le système est donc bien homomorphique.

Ce système est donc très attractif. En effet, son coût est très faible et sa sécurité sémantique repose sur un problème bien connu. Malheureusement, son expansion est catastrophique, puisque seulement un bit est chiffré par un élément de $\mathbf{Z}/n\mathbf{Z}$ où n est un entier RSA, *i.e.*, l'expansion est de $|n|_2$.

Le système de Benaloh (1988)

Le point négatif du système précédent est cette très grande expansion. Pour agrandir l'espace des messages clairs, il convient d'avoir à sa disposition un groupe quotient d'ordre supérieur. C'est la démarche de Benaloh dans sa thèse ([Ben88]).

Cadre : Le système que propose Benaloh rentre dans le cadre de la fonction trappe présentée en sous-section II–1.2. On pose toujours $\mathrm{G} = (\mathbf{Z}/n\mathbf{Z})^\times$, et l'entier k est un nombre premier tel que k divise $\varphi(n) = |\mathrm{G}|$ et tel que k ne divise pas $\varphi(n)/k$. L'entier k est donc impair, et on a en fait une généralisation du système de Goldwasser-Micali.

D'après le corollaire II – 2, page 16, le groupe quotient G/G^k est alors d'ordre k. Du fait que k est choisi premier, tout élément g de $\mathrm{G} \setminus \mathrm{G}^k$ donnera un générateur $\overline{g}$ de G.

Chiffrement : On utilise la fonction de chiffrement suivante :

$$\mathcal{E}_{(\mathbf{Z}/n\mathbf{Z})^\times,k,g} : \begin{cases} \{0,1,\ldots,k-1\} \times (\mathbf{Z}/n\mathbf{Z})^\times & \longrightarrow & (\mathbf{Z}/n\mathbf{Z})^\times \\ (m \quad , \quad r) & \longmapsto & g^m r^k \end{cases}$$

où m est le message à chiffrer et r est choisi au hasard.

Cette fonction est surjective , mais ce n'est toujours pas un morphisme à moins que g soit d'ordre k dans $(\mathbf{Z}/n\mathbf{Z})^\times$.

Sécurité : La fonction de déchiffrement est le calcul du logarithme en base $\overline{g}$ dans G/G^k. La sécurité du système est donc basée sur le problème de classe de résidualité d'ordre k (cf. définition II – 5).

D'après le théorème II – 11, la sécurité sémantique repose sur le problème de résidualité d'ordre k dans G, *i.e.*, le problème de reconnaissance des puissances k–ièmes, avec ici $G = (\mathbf{Z}/n\mathbf{Z})^\times$ et k un nombre premier. Dans la section 2.8 de [Ben88], on montre que si l'on peut résoudre ce problème pour quelques instances, alors, en temps polynomial, on peut le résoudre pour toutes les instances.

Pour argumenter sur la difficulté de ce problème, Benaloh renvoie à un résultat d'Adelman et McDonnell (cf. [AD82]) qui montre qu'à partir d'un oracle résolvant ce problème pour k variable, on peut construire un algorithme efficace (mais non polynomial) factorisant n.

Pour finir, comme dans le cas quadratique, calculer une racine k–ième permet de factoriser n. En effet, supposons que l'on dispose d'un oracle nous donnant une racine k–ième d'un élément de $(\mathbf{Z}/n\mathbf{Z})^\times$. On choisit x au hasard dans $(\mathbf{Z}/n\mathbf{Z})^\times$ et on calcule x^k. L'oracle nous retourne un élément y de $(\mathbf{Z}/n\mathbf{Z})^\times$ tel que $y^k = x^k$. Du choix de k (k est premier, k divise $\varphi(n)$ et k ne divise par $\varphi(n)/k$), on déduit que l'on a, par exemple, $k \mid (p-1)$ et $k \nmid (q-1)$. On a alors une seule racine k–ième modulo q. Ainsi, si $x \neq y$, on a

$$x \not\equiv y \pmod{p} \qquad \text{et} \qquad x \equiv y \pmod{q},$$

donc $\operatorname{pgcd}(x-y, n) = q$. Remarquons que comme on a k racines k–ièmes modulo n, l'oracle nous renvoie un y non congru à x avec probabilité $1 - 1/k$.

Déchiffrement : Soit c un chiffré. Voyons maintenant comment calculer $[\![c]\!]_g$ connaissant la factorisation de n. D'après le lemme II – 3,

$$c^{\varphi(n)/k} = g^{m\varphi(n)/k},$$

et on est ramené à un calcul de logarithme discret dans g. Benaloh propose de se ramener à un calcul de logarithme discret dans le sous-groupe des racines k–ièmes de l'unité de G, noté $\mathcal{U}_k$. En effet, d'après la remarque page 17, $\mathcal{U}_k$ est isomorphe à G/G^k. On note B un entier obtenu par la relation de Bézout, $Ak - B\varphi(n)/k = -1$. Étant donné $c \in (\mathbf{Z}/n\mathbf{Z})^\times$,

$$\xi \;:=\; c^{B\varphi(n)/k}$$

est l'unique racine k–ième de l'unité telle que $[\![c]\!]_g = [\![\xi]\!]_g$.

On pré-calcule l'élément ξ de $\mathcal{U}_k$ tel que $\xi \equiv g \pmod{\mathrm{G}^k}$, *i.e.*, tel que l'on ait les congruences $[\![\xi]\!]_g \equiv 1 \pmod{k}$. Par l'isomorphisme entre G/G^k et $\mathcal{U}_k$, l'élément ξ est un générateur de $\mathcal{U}_k$ et pour tout $i \in \{0, \ldots, k-1\}$,

$$\left[\!\left[\xi^i\right]\!\right]_g \equiv i \pmod{k}.$$

Pour déchiffrer un message c, on cherche la racine k–ième de l'unité équivalente et on regarde quel est son logarithme discret dans une table.

Tout cela nécessite de faire des pré-calculs en $\mathcal{O}(k)$ puis une recherche dans une table de k éléments. Benaloh propose une version améliorée de cet algorithme en utilisant une méthode de type « Baby Step - Giant Step », on obtient ainsi des pré-calculs en $\mathcal{O}\left(\sqrt{k}\,|k|_2\right)$, puis une recherche en temps similaire. Cet algorithme de déchiffrement peu performant impose de prendre un k de taille modérée. Par conséquent, l'expansion du cryptosystème qui est de $|n|_2 / |k|_2$ reste importante.

Le système de Naccache-Stern (1998)

Cadre : Dans [NS98], Naccache et Stern considèrent toujours le groupe $\mathrm{G} := (\mathbf{Z}/n\mathbf{Z})^\times$. Leur système est une amélioration du précédent. Les auteurs n'utilisent non pas un k premier petit, mais un k grand, sans carré, et B–friable où B est de l'ordre de 10 bits. Cela permet d'avoir une expansion plus petite tout en ayant un déchiffrement rapide par utilisation des restes chinois.

Voyons cela plus en détail. On prend toujours k tel que $k \mid \varphi(n)$ et tel que k soit premier avec $\varphi(n)/k$, de telle sorte que le groupe quotient G/G^k soit d'ordre k. Reste à trouver un générateur de G. D'après le lemme II – 4, tout élément g de $(\mathbf{Z}/n\mathbf{Z})^\times$ dont l'ordre est un multiple de k donnera un générateur. On renvoie le lecteur à [NS98] pour la construction d'un tel élément. Le système est alors identique à celui de Benaloh, seul le déchiffrement va différer.

Déchiffrement : Voyons maintenant comment retrouver $m := [\![c]\!]_g$ étant donné un élément g de G. On utilise la méthode de Pohlig-Hellman. On note $k = \prod_{i \in \mathrm{I}} p_i$ la décomposition en facteurs premiers de k. On va calculer $m_i := m \bmod p_i$. On note l_i le quotient de la division euclidienne de m par p_i. Comme $\frac{\varphi(n)}{p_i}$ est un multiple de $\frac{\varphi(n)}{k}$, on a

$$c^{\frac{\varphi(n)}{p_i}} = g^{m\frac{\varphi(n)}{p_i}}.$$

Or, on a aussi

$$g^{m\frac{\varphi(n)}{p_i}} = g^{\frac{(m_i + l_i p_i)\varphi(n)}{p_i}} = g^{\frac{m_i \varphi(n)}{p_i}} \times g^{l_i \varphi(n)} = g^{m_i \frac{\varphi(n)}{p_i}}.$$

Comme les p_i sont petits (car k est B–friable), m_i peut être trouvé par recherche exhaustive en calculant

$$g^{l\frac{\varphi(n)}{p_i}}$$

pour $l = 1, 2, \ldots, p_i - 1$. On récupère ainsi m_i modulo p_i pour tous les facteurs de k. Par le théorème chinois, on en déduit m modulo k.

Naccache et Stern annoncent une complexité en $|n|_2^5 \log(|n|_2)$. Comparé au système de Benaloh, l'expansion est aussi améliorée, comme k peut être choisi grand. D'après la discussion sur le choix des paramètres faite dans [NS98], on peut espérer avoir une expansion de l'ordre de 4.

Le système de Okamoto-Uchiyama (1998)

Pour améliorer les cryptosystèmes précédents, Okamoto et Uchiyama ont eu l'idée dans [OU98a] de changer de groupe de base : ils posent $n = p^2 q$ où p et q sont toujours deux grands nombres premiers. En posant $\mathrm{G} := \left(\mathbf{Z}/p^2\mathbf{Z}\right)^\times$, leur cryptosystème utilise le groupe quotient G/G^p, *i.e.*, $k = p$. Le système de Paillier qui suit est une amélioration directe de ce cryptosystème (l'expansion est de 2 au lieu de 3 pour Okamoto-Uchiyama). L'avantage de ce système par rapport à celui de Paillier est que les auteurs montrent que la sécurité de leur système est équivalente au problème de la factorisation de n. Malheureusement, une attaque à textes chiffrés choisis permet de retrouver cette factorisation.

Le système de Paillier (1999)

Cadre : Paillier, dans la lignée des idées d'Okamoto et Uchiyama, reprend, comme module n, un entier RSA, $n = pq$, puis considère le groupe $\mathrm{G} := \left(\mathbf{Z}/n^2\mathbf{Z}\right)^\times$ (cf. [Pai99]). Il utilise ensuite $k = n$. Ceci va permettre d'améliorer tous les systèmes précédents en ayant, d'une part, une expansion égale à 2, et d'autre part, en faisant apparaître un élément g d'ordre n tel que le logarithme discret en base g soit facile. On se retrouve donc pleinement dans le cadre des hypothèses de la sous-section II–1.2.

Le groupe G est d'ordre $\varphi(n^2) = n\varphi(n)$. En supposant que n est premier avec $\varphi(n)$, on peut appliquer le corollaire II – 2 dans $\mathbf{Z}/n^2\mathbf{Z}$. On en déduit alors que le groupe quotient G/G^n est d'ordre n.

On note λ l'indicateur de Carmichael. On a $\lambda(n) = \mathrm{ppcm}(p-1, q-1)$.Comme n est premier avec $\varphi(n)$, on a de plus $\lambda(n^2) = n\lambda(n)$. L'entier n divise donc $\lambda(n^2)$ et n est premier avec $\lambda(n^2)/n$. Une généralisation directe du lemme II – 3 nous donne

$$\mathrm{G}^n = \left\{ x \in \mathrm{G},\ x^{\lambda(n)} = 1 \right\}.$$

Élément d'ordre n : En prenant un élément g de G d'ordre divisible par n, Paillier obtient un générateur de G/G^n, d'après le lemme II – 4. Cependant, comme noté par Paillier (cf. "Lemma 4" de [Pai99]), le problème $\mathrm{Classe}_{\mathrm{G},k}$, sur lequel repose la sécurité du système, ne dépend pas du choix de g dans l'ensemble des éléments d'ordres divisibles par n. On choisit donc pour g un élément d'ordre n. Ceci va non seulement simplifier la description, mais de plus diminuer son coût calculatoire.

D'après la remarque page 17, un élément d'ordre n sera un générateur du sous-groupe cyclique des racines n–ièmes de l'unité de G, noté $\mathscr{U}_n$. Le noyau de la surjection canonique de $\mathrm{G} = \left(\mathbf{Z}/n^2\mathbf{Z}\right)^\times$ sur $(\mathbf{Z}/n\mathbf{Z})^\times$ est aussi d'ordre n.

Par définition des racines n–ièmes de l'unité, ces deux sous-groupes d'ordre n coïncident. On a donc $\mathrm{G}/\mathscr{U}_n \cong (\mathbf{Z}/n\mathbf{Z})^\times$ et on a l'expression

$$\mathscr{U}_n = \left\{x \in \mathrm{G},\, x \equiv 1 + bn \pmod{n^2},\, b \in \mathbf{Z}/n\mathbf{Z}\right\}.$$

Ainsi, un élément g de G, tel que $g \equiv 1+n \pmod{n^2}$ sera un élément d'ordre n de G, car pour tout entier l tel que $0 < l < n-1$, on a

$$(1+n)^l \equiv 1 + ln \not\equiv 1 \pmod{n^2}.$$

Ainsi, le logarithme discret en base g dans $\langle g\rangle = \mathscr{U}_n$ sera très facile à calculer, car vu le calcul précédent,

$$\log_{1+n}(u) \equiv u_1, \quad \forall u \in \mathscr{U}_n,$$

en utilisant la représentation (u_0, u_1) de u définie en section III–1.

Les puissances n–ièmes : Pour générer des puissances n–ièmes, on va utiliser la première méthode donnée en page 22. C'est à dire utiliser l'isomorphisme

$$(\mathbf{Z}/n\mathbf{Z})^\times \longrightarrow \mathrm{G}^n, \quad r \longmapsto r^n.$$

En pratique, on va tirer des éléments r_0 dans $\{1, \ldots, n-1\}$ qui, comme n est un entier RSA, auront une probabilité proche de 1 d'être premiers avec n. On calcule alors r^n dans G où r est l'élément de G de représentation $(r_0, 0)$. Il n'y a donc ici aucun problème pour relever les éléments de $(\mathbf{Z}/n\mathbf{Z})^\times$ dans $\left(\mathbf{Z}/n^2\mathbf{Z}\right)^\times$.

Chiffrement : La fonction de chiffrement est donc l'isomorphisme

$$\mathscr{E}_{(\mathbf{Z}/n^2\mathbf{Z})^\times, n, g} : \begin{cases} \mathbf{Z}/n\mathbf{Z} \times (\mathbf{Z}/n\mathbf{Z})^\times & \xrightarrow{\sim} & \left(\mathbf{Z}/n^2\mathbf{Z}\right)^\times \\ (m\ ,\ r) & \longmapsto & g^m r^n \end{cases}$$

où m est le message à chiffrer et r est choisi au hasard.

Déchiffrement : Passons maintenant au déchiffrement. On procède comme indiqué en section II–1.2. Soit c un chiffré de m, on cherche à calculer $[\![c]\!]_g = m$. La trappe est l'entier $\lambda(n)$. Le calcul $c^{\lambda(n)}$ donne $g^{m\lambda(n)}$. Le calcul du logarithme en base g dans $\langle g\rangle$ donne alors $m\lambda(n)$ dans $\mathbf{Z}/n\mathbf{Z}$ puis m comme $\lambda(n)$ est premier avec n. Notons que la connaissance de la trappe est équivalente à la connaissance de la factorisation de n.

Efficacité : Le coût du chiffrement est celui d'une exponentiation à la puissance n modulo n^2 et d'une multiplication modulo n (en utilisant la représentation en base n, la multiplication de g^m par r^n ne requiert qu'une multiplication modulo n).

Le déchiffrement requiert une exponentiation modulo n^2 à la puissance $\lambda(n)$ et une multiplication modulo n. Dans [Pai99], Paillier montre comment faire un déchiffrement optimisé par l'utilisation des restes chinois.

L'expansion E est maintenant égale à 2.

Sécurité : La sécurité du système est basée sur le problème de classe de résidualité d'ordre n dans $(\mathbf{Z}/n^2\mathbf{Z})^\times$ où n est un entier RSA, noté $\text{Classe}_{(\mathbf{Z}/n^2\mathbf{Z})^\times,n}$.

Par le corollaire II – 9, comme le problème du calcul de l'ordre de G est équivalent à celui de la factorisation de n (noté FACT(n)), on a la hiérarchie de problèmes :

$$\text{Classe}_{(\mathbf{Z}/n^2\mathbf{Z})^\times,n} \overset{\mathscr{P}}{\Longleftarrow} \text{RSA}_{n,n} \overset{\mathscr{P}}{\Longleftarrow} \text{FACT}(n),$$

en notant plus simplement $\text{RSA}_{n,n}$ le problème $\text{RSA}_{(\mathbf{Z}/n\mathbf{Z})^\times,n}$ introduit par la définition II – 7.

Le théorème II – 15 obtenu par Catalano, Nguyen et Stern dans [CNS02] permet de préciser ces réductions. Pour bien voir cela, on va définir deux problèmes, cas particulier du problème $\text{Hensel}_{G,g}-f$ défini dans la définition II – 13.

Avec les notations de la section II–2.1, on a $\Omega := G$, $\overline{\Lambda} := (\mathbf{Z}/n\mathbf{Z})^\times$ et

$$\Lambda := \{x \in \mathbf{N},\ 1 < x < n,\ \text{pgcd}(x,n) = 1\}.$$

La fonction f est la fonction de Λ dans Ω qui à un élément x associe $(x^n \bmod n^2)$. On note alors H–$\text{RSA}_{n,2}$ le problème $\text{Hensel}_{G,g}-f$ défini dans la définition II – 13, *i.e.*, étant donné un élément c de $\overline{\Lambda}$, le problème de trouver la valeur $(r^n \bmod n^2)$, en notant r l'élément de Λ tel que $c \equiv r^n \pmod n$.

De même, si f est maintenant la fonction de Λ dans l'ensemble sous-jacent de $(\mathbf{Z}/n^3\mathbf{Z})^\times$ qui à un élément x associe $(x^n \bmod n^3)$, le problème $\text{Hensel}_{G,g}-f$ devient : étant donné un élément c de $\overline{\Lambda}$, trouver la valeur $(r^n \bmod n^3)$, en notant r l'élément de Λ tel que $c \equiv r^n \pmod n$. On note H–$\text{RSA}_{n,3}$ ce problème.

Les théorèmes II – 14 et II – 15 donnent alors

$$\begin{array}{ccc} \text{Classe}_{(\mathbf{Z}/n^2\mathbf{Z})^\times,n} & \overset{\mathscr{P}}{\Longleftarrow} & \text{RSA}_{n,n} \\ \Updownarrow_{\mathscr{P}} & & \Updownarrow_{\mathscr{P}} \\ \text{H–RSA}_{n,2} & \overset{\mathscr{P}}{\Longleftarrow} & \text{H–RSA}_{n,3} \end{array}$$

ce qui laisse à penser que les problèmes $\text{Classe}_{(\mathbf{Z}/n^2\mathbf{Z})^\times,n}$ et $\text{RSA}_{n,n}$ sont bien distincts.

La sécurité sémantique est équivalente au problème de résidualité d'ordre n dans le groupe $(\mathbf{Z}/n^2\mathbf{Z})^\times$, où n est un entier RSA. On note ce problème $\text{Rés}_{(\mathbf{Z}/n^2\mathbf{Z})^\times,n}$.

Le système de Damgård et Jurik (2003)

Dans [DJ01], les auteurs proposent une généralisation du système de Paillier dans les groupes $(\mathbf{Z}/n^{s+1}\mathbf{Z})^\times$, où $s > 0$. Avec $s = 1$, on retrouve le système de Paillier. Cette généralisation permet de diminuer l'expansion en faisant grandir s. Elle offre aussi de multiples applications : ajustement de la taille des messages clairs, cryptographie de seuil et vote électronique (cf. [DJ01]).

Cadre : On suppose toujours que $\operatorname{pgcd}(n,\varphi(n))=1$. On considère $G := \left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times$ et $k := n^s$. On a $\varphi(n^{s+1}) = n^s\varphi(n)$. Par le corollaire II-2, le groupe quotient G/G^k est d'ordre n^s. D'autre part, comme on a $\operatorname{pgcd}\big(n,\varphi(n)\big)=1$, on a $\lambda\big(n^{s+1}\big) = n^s\lambda(n)$. On a donc encore la caractérisation

$$G^{n^s} = \left\{z \in G,\, z^{\lambda(n)} = 1\right\}.$$

On prend toujours un générateur dans le noyau de la surjection canonique du groupe $\left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times$ sur le groupe $(\mathbf{Z}/n\mathbf{Z})^\times$. En effet, ce noyau est toujours le sous-groupe cyclique d'ordre n^s des racines n^s-ièmes de l'unité. Damgård et Jurik montrent que, si s est inférieur à $\min(p,q)$, un élément g de G tel que $g \equiv 1+n \pmod{n^{s+1}}$ est d'ordre n^s dans $\left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times$.

Les puissances n^s-ièmes sont toujours produites grâce à l'isomorphisme

$$(\mathbf{Z}/n\mathbf{Z})^\times \longrightarrow G^{n^s}, \quad r \longmapsto r^{n^s}.$$

Remarquons que l'on peut également publier une puissance n^s-ième comme vu en page 23 pour éviter d'avoir à faire une exponentiation aussi lourde.

Chiffrement : La fonction de chiffrement est l'isomorphisme

$$\mathscr{E}_{\left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times, n^s, g} : \begin{cases} \mathbf{Z}/n^s\mathbf{Z} \times (\mathbf{Z}/n\mathbf{Z})^\times & \xrightarrow{\sim} \left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times \\ (m \quad , \quad r) & \longmapsto \quad g^m r^{n^s} \end{cases}$$

où m est le message à chiffrer et r est choisi au hasard.

Déchiffrement : Pour déchiffrer, on se ramène toujours, par exponentiation à la puissance $\lambda(n)$, à un calcul de logarithme discret en base g dans $\langle g\rangle$. La situation est plus complexe que dans le cas $s=1$ car on a maintenant la congruence suivante :

$$(1+n)^m \equiv 1 + mn + \binom{m}{2}n^2 + \cdots + \binom{m}{s}n^s \pmod{n^{s+1}},$$

si $m \geq s$. Damgård et Jurik donnent un algorithme qui calcule récursivement $(m \bmod n^k)$ pour k allant de 1 à s, similaire à celui donné dans les corps quadratiques (cf. figure III.5 page 61).

Sécurité : La sécurité du système est basée sur le problème de classe de résidualité d'ordre n^s dans $\left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times$ où n est un entier RSA, noté $\text{Classe}_{\left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times, n}$.

On relie ce problème au cas $s=1$. Soit c un élément de $\left(\mathbf{Z}/n^2\mathbf{Z}\right)^\times$ chiffré d'un élément m de $\mathbf{Z}/n\mathbf{Z}$ dans le cas $s=1$. En voyant c comme un élément de $\left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times$ avec $s>1$, c est un chiffré d'un message m' de $\mathbf{Z}/n^s\mathbf{Z}$. On voit facilement que $m' \equiv m \pmod{n}$. Ceci permet à Damgård et Jurik de prouver la réduction, valable pour tout entier naturel s non nul :

$$\text{Classe}_{\left(\mathbf{Z}/n^2\mathbf{Z}\right)^\times, n} \overset{\mathscr{P}}{\Longleftarrow} \text{Classe}_{\left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times, n}. \tag{IV.2}$$

Le même type de réductions vues dans le cas $s = 1$ faisant intervenir le problème H–RSA sont toujours valables dans le cas $s > 1$.

Damgård et Jurik montrent également que la sécurité sémantique de leur système est équivalente à celle du système de Paillier, *i.e.*, que pour tout entier naturel s non nul

$$\text{Rés}_{(\mathbf{Z}/n^2\mathbf{Z})^\times,n} \overset{\mathscr{P}}{\Longleftrightarrow} \text{Rés}_{(\mathbf{Z}/n^{s+1}\mathbf{Z})^\times,n^s}.$$

Efficacité : Au final, l'expansion du cryptosystème est de $1+1/s$ et peut donc être rendue proche de 1 pour s grand. Cependant, faire croitre s rend le chiffrement et le déchiffrement plus long.

On résume les caractéristiques des systèmes de Paillier et de Damgård-Jurik dans le tableau suivant :

Cryptosystème	Paillier	Damgård-Jurik
Cadre	$(\mathbf{Z}/n^2\mathbf{Z})^\times$	$(\mathbf{Z}/n^{s+1}\mathbf{Z})^\times$, $s>1$
Taille de l'entrée	$\lvert n\rvert_2$	$s\lvert n\rvert_2$
Taille de la sortie	$2\lvert n\rvert_2$	$(s+1)\lvert n\rvert_2$
Expansion	2	$1+1/s$
Coût du chiffrement	$\frac{9}{2}\lvert n\rvert_2$	$\frac{3}{4}s(s+1)(s+2)\lvert n\rvert_2$
Coût de déchiffrement	$\frac{3}{2}\lvert n\rvert_2$	$\frac{1}{4}(s+1)(s+2)\lvert n\rvert_2$

L'unité du coût des opérations est le coût de la multiplication modulo n. Dans le calcul du coût, on a compté seulement les opérations les plus importantes, *i.e.*, les exponentiations modulaires. Les restes chinois sont utilisés pour le déchiffrement. En suivant les résultats de la section III–1, une multiplication modulo n^{s+1} avec $s > 0$ est estimée aussi coûteuse que $(s+1)(s+2)/2$ multiplications modulo n. Le coût d'une multiplication modulo p^{s+1} est estimé égal à celui de $(s+1)(s+2)/6$ multiplications modulo n.

Signalons que le système de Damgård-Jurik améliore l'expansion mais ne diminue pas les coûts : pour chiffrer ou déchiffrer k blocs de $|n|$ bits, k applications du système de Paillier sont plus efficaces qu'une seule du système de Damgård-Jurik avec $s = k$, et ce pour tout entier positif k.

2.2. Dans les courbes elliptiques, le système de Galbraith

Dans cette sous-section, on adapte dans les courbes elliptiques le cryptosystème de Paillier généralisé par Damgård et Jurik. Pour cela, on va encore utiliser la fonction trappe définie en sous-section II–1.2. Cette adaptation a été faite par Galbraith dans [Gal02].

Cadre

On note toujours $n := pq$ un entier RSA et s un entier naturel non nul. Le cas $s = 1$ correspondra à l'adaptation du système de Paillier. Soient a et b deux éléments de $\mathbf{Z}/n^{s+1}\mathbf{Z}$

tels que $4a^3+27b^2$ soit inversible. On va utiliser le groupe $G := E_{n^{s+1}}(a,b)$ étudié en section III–2.

Avec les notations de la section II–1, on pose $k := n^s$, c'est à dire que l'on va utiliser le groupe quotient G/G^{n^s}, où G^{n^s} sera le sous-groupe de G constitué des éléments pouvant s'écrire $n^s P$ où $P \in G$. Ces éléments seront par la suite toujours appelés « puissances n^s–ièmes » par abus de langage. On note $\mu := \mathrm{ppcm}\left(\left|E_p\right|, \left|E_q\right|\right)$ et on suppose que μ et n sont premiers entre eux. On a vu en sous-section III–2.5 que le groupe $G = E_{n^{s+1}}$ est d'ordre $n^s\left|E_n\right|$. De plus, pour tout élément P de G, on a $(n\mu)P = \mathcal{O}$, en notant $\mathcal{O}$ l'élément neutre de G.

Comme n^s divise l'ordre de G et comme n^s est premier avec $|G|/n^s$, le groupe quotient G/G^{n^s} est d'ordre n^s (cf. corollaire II – 2). De plus, par le lemme II – 3,

$$G^{n^s} = \left\{P \in E_{n^{s+1}},\ \mu P = \mathcal{O}\right\}.$$

Élément d'ordre n

Pour retrouver le contexte de la fonction trappe homomorphique définie au chapitre II, il nous reste à trouver un générateur de G/G^{n^s}. Pour cela, d'après le lemme II – 4, il suffit de trouver un élément de $E_{n^{s+1}}$ d'ordre n^s. En suivant la remarque de la page 17, on cherche ce générateur dans le sous-groupe d'ordre n^s des racines n^s–ièmes de l'unité de G (toujours par abus de langage, on nomme ainsi les éléments P de G tels que $n^s P = \mathcal{O}$).

On retrouve alors la situation rencontrée dans les quotients de $\mathbf{Z}$. Le noyau de la surjection canonique de $G = E_{n^{s+1}}$ sur E_n est le sous-groupe E_1 de G constitué par les éléments au-dessus du point à l'infini $(0:1:0)$ de E_n. On a vu en sous-section III–2.5 que ce groupe est d'ordre n^s. Le sous-groupe des racines n^s–ièmes de l'unité est donc égal à E_1. On cherche alors un générateur de E_1.

On a vu, toujours en sous-section III–2.5, une description des éléments de E_1 ainsi que la loi de groupe permettant d'additionner ces éléments. On distingue deux cas :

Pour $s < 5$ ou si $a = 0$ pour $s < 7$, cette loi, donnée par l'expression (III.6) page 40, est une simple addition dans $\mathbf{Z}/n^{s+1}\mathbf{Z}$. On établit ainsi un isomorphisme explicite entre $\mathbf{Z}/n^s\mathbf{Z}$ et E_1 :

$$\begin{array}{ccc} \mathbf{Z}/n^s\mathbf{Z} & \xrightarrow{\sim} & E_1 \\ k & \longmapsto & (kn:1:w(kn)) \end{array}$$

où l'expression de w est donnée par (III.5) page 40.

Pour tout $m \in \mathbf{Z}/n^s\mathbf{Z}$, on note $\mathcal{O}_m :\equiv (mn:1:w(kn)) \equiv m\mathcal{O}_1 \pmod{n^s+1}$. Par l'isomorphisme, on décrit ainsi tous les éléments de E_1. L'élément $g := \mathcal{O}_1$ fournira donc un générateur du groupe quotient G/G^{n^s}.

Le logarithme discret dans E_1 en base g sera immédiat à calculer vu que la coordonnée X de $\mathcal{O}_m$ est mn.

Dans le cas général, la loi du groupe formel est beaucoup plus complexe. On peut cependant voir que $(n:1:w(n))$, toujours noté $\mathcal{O}_1$, est d'ordre n^s quel que soit s (cf. [Gal02]). Cet élément fournira donc un générateur de G/G^{n^s}. Pour coïncider avec les notations prises dans l'autre cas, on note pour tout $m \in \mathbb{Z}_{n^s}$, $\mathcal{O}_m := m\mathcal{O}_1$, calculé avec la loi du groupe formel (III.6). Notons bien que dans ce cas $\mathcal{O}_m \neq (mn:1:w(mn))$.

Le logarithme discret dans E_1 en base g sera plus compliqué à calculer. Soit $\mathcal{O}_m$ un élément de E_1, on cherche à retrouver m. On procède de la manière suivante : on écrit m en base n^4 : $m \equiv m_0 + m_1 n^4 + m_2 n^8 + \dots \pmod{n^s}$. On retrouve ensuite les m_i par récurrence. En réduisant $\mathcal{O}_m$ modulo n^5, on récupère $\mathcal{O}_{m_0}$ dans E_{n^5}, on peut donc en déduire directement m_0. Pour retrouver m_1, on soustrait $\mathcal{O}_{m_0}$ à $\mathcal{O}_m$ dans $E_{n^{s+1}}$, ceci nous donne, par définition, $\mathcal{O}_{m_1 n^4 + m_2 n^8 + \dots}$. En réduisant ce résultat modulo n^9, on obtient $\mathcal{O}_{m_1 n^4}$. De part la forme de la loi du groupe formel, la coordonnée en X de ce point est égale à $n^5 m_1$. On trouve ainsi m_1. Par itération de ce procédé on récupère m dans $\mathbb{Z}_{n^s}$ (cf. [Gal02]).

Chiffrement

On construit donc un cryptosystème basé sur le calcul du logarithme discret dans le groupe quotient G/G^{n^s}. On chiffre des messages de $\mathbb{Z}_{n^s}$ par l'application

$$m \longmapsto C := \mathcal{O}_m + P,$$

où P est une puissance n^s-ième. Le calcul de $\mathcal{O}_m := m\mathcal{O}_1$ se fait par la loi du groupe formel ou directement si $s < 5$. Si l'on souhaite travailler en coordonnées affines, l'addition entre $\mathcal{O}_m$ et P peut se faire par la formule (III.11) page 44, si P n'est pas au-dessus d'un point d'ordre deux, sinon par la formule (III.14) page 47. En coordonnées projectives, on utilisera la formule (III.12) page 46.

Déchiffrement

Pour retrouver m, on calcule

$$\mu C = \mu \mathcal{O}_m = \mathcal{O}_{m\mu \bmod n^s}.$$

La connaissance de la trappe μ permet donc de réduire le calcul du logarithme discret dans G/G^k au calcul du logarithme discret dans E_1 en base $g = \mathcal{O}_1$. On a vu en page 37 que le problème du calcul de μ était équivalent à celui de la factorisation de n.

Les puissances n^s–ièmes

Pour pouvoir chiffrer, il reste à savoir générer efficacement une puissance n^s–ième dans $E_{n^{s+1}}$. On utilise la seconde solution donnée page 23. En effet, comme vu en sous-section III–2.2, il semble difficile sans connaître la factorisation de n de construire un point de G. De plus, si on veut construire un système homomorphique, on ne peut utiliser des groupes variables puisque les chiffrés doivent être dans le même groupe.

On reste sur une courbe fixe, *i.e.*, a et b sont deux éléments fixés, et on publie une puissance n^s-ième Q de $G = E_{n^{s+1}}(a, b)$ (on peut obtenir Q en prenant un point quelconque de $E_{n^{s+1}}$ et en le mettant à la puissance n^s). Le chiffreur génère alors d'autres puissances n^s-ièmes en calculant kQ où k est pris inférieur à n. Si on note ℓ l'ordre de Q, la fonction de chiffrement est le morphisme injectif

$$\mathcal{E}_{E_{n^{s+1}}, n^s, g, Q} : \begin{cases} \mathbf{Z}/n^s\mathbf{Z} \times \mathbf{Z}/\ell\mathbf{Z} & \longrightarrow & G \\ (m \ , \ r) & \longmapsto & m.\mathcal{O}_1 + r.Q \end{cases}$$

Sécurité

La sécurité sémantique du système construit selon ce procédé repose sur la difficulté du problème de reconnaissance des éléments du sous-groupe de $E_{n^{s+1}}$ engendré par Q. Il convient donc d'avoir un point Q d'ordre grand (l'ordre de Q divise μ).

Côté sécurité, le cryptosystème est basé sur le calcul du logarithme discret dans G/G^{n^s}. Ce problème se réduit au problème $KMOV_{n,n^s}$ et à celui de la factorisation de n. On obtient de plus des résultats similaires à ceux montrer par Damgård et Jurick, à savoir le système avec $s > 1$ sera plus sûr que celui avec $s = 1$.

Remarquons que l'on doit travailler avec un entier n de taille aussi grande que dans les quotients de $\mathbf{Z}$, contrairement aux cryptosystèmes basés sur le logarithme discret dans E_p où l'on peut utiliser des entiers p de tailles 10 fois plus faibles que pour les mêmes cryptosystèmes dans $\mathbf{Z}/p\mathbf{Z}$.

Efficacité

Étudions le coût algorithmique de ce système. Pour le chiffrement, l'étape la plus importante est le calcul de r.Q. On utilise les coordonnées affines et un algorithme "double and add" avec les formules classiques. Pour l'addition, on utilise la formule (III.9) page 43, avec l'expression (III.8) de λ, ce qui donne 3 multiplications et une inversion modulo n^{s+1}. Avec les estimations de la sous-section III–1 cela donne un coût de

$$(2s^2 + 6s + 3)\mathrm{M} + \mathrm{I},$$

où M et I désignent respectivement le coût de la multiplication et de l'inversion modulo n. Pour le double, on trouve, avec la formule (III.9) utilisant cette fois ci l'expression (III.10) de λ, un coût de

$$\frac{5s^2 + 15s + 8}{2}\mathrm{M} + \mathrm{I}.$$

Ainsi le calcul de r.Q avec $r < n$ prend au maximum, en moyenne sur les entiers n utilisés,

$$\frac{7s^2 + 21s + 11}{2}\,|n|_2\,\mathrm{M} + \frac{3}{2}\,|n|_2\,\mathrm{I}.$$

Pour le déchiffrement, l'étape la plus longue est l'exponentiation à la puissance μ dans G. En procédant avec les restes chinois on obtient un coût de

$$\frac{7s^2 + 21s + 11}{6}\,|n|_2\,\mathrm{M} + \frac{1}{2}\,|n|_2\,\mathrm{I},$$

en considérant qu'une multiplication modulo p (resp. une inversion modulo p) coûte M/3 (resp. I/3) et que $|E_p|$ et $|E_q|$ sont de tailles $|n|_2/2$.

Contrairement à ce que l'on a vu dans les quotients de $\mathbf{Z}$, augmenter s peut permettre de diminuer le coût : pour chiffrer ou déchiffrer k blocs de $|n|$ bits, k applications du système avec $s = 1$ sont moins efficaces qu'une seule du système avec $s = k$, dés que $k > 6$. Cela est dû au fait que la taille de l'exposant des exponentiations est fixe et à la présence des inversions modulaires dans les formules d'addition et de double.

Expansion

En augmentant s, on diminue l'expansion. Cette expansion est *a priori* de $2+2/s$ puisqu'un point C de $\mathrm{E}_{n^{s+1}}$ est codé à l'aide de $2(s+1)|n|$ bits. Ceci peut être réduit en stockant le chiffré C de manière efficace. Un point P de $\mathrm{E}_{n^{s+1}}$ peut être codé sous la forme $(\widetilde{\mathrm{P}}, k)$ où $\widetilde{\mathrm{P}}$ est le point de E_n au dessous de P et $k \in \mathbf{Z}/n^s\mathbf{Z}$ est tel que $\mathrm{P} = (\widetilde{\mathrm{P}})_l + \mathcal{O}_k$ où $(\widetilde{\mathrm{P}})_l$ désigne le relèvement de $\widetilde{\mathrm{P}}$, défini par la proposition ci-après. Ce stockage occupe alors $(2+s)|n|$ bits, l'expansion est alors réduite à $1+2/s$.

On définit donc une manière de relever les éléments de E_n. On utilise le lemme de Hensel, de manière similaire à la proposition III – 9, énoncée dans le cadre des corps quadratiques.

Proposition IV – 6 (Relèvement des points de E_n dans $\mathrm{E}_{n^{s+1}}$)**.**

Soient a et b deux entiers tel que $4a^3+27b^2$ soit premier avec n. On considère un point P *de la courbe* $\mathrm{E}_n\big(a \pmod n, b \pmod n\big)$ *que l'on veut relever modulo n^{s+1} dans la courbe* $\mathrm{E}_{n^{s+1}}\big(a \pmod{n^{s+1}}, b \pmod{n^{s+1}}\big)$. *On note f le polynôme de* $\mathbf{Z}[\mathrm{X}]$ *associé à l'équation affine de ces courbes :*

$$f(x,y) := y^2 - (x^3 + ax + b).$$

On va procéder par récurrence en utilisant le lemme de Hensel. Supposons que l'on ait déjà relevé P *dans* E_{n^k} *pour un k tel que $1 \leq k \leq s$. On écrit* P *sous la forme (x,y) avec x et y des entiers définis modulo n^k. On a deux cas :*

— *si* $\mathrm{pgcd}(y,n) = 1$, *on relève* P *dans* $\mathrm{E}_{n^{k+1}}$ *en le point de coordonnées (x,y') où*

$$y' \equiv y - \big((2y)^{-1} \bmod n\big) f(x,y) \pmod{n^{k+1}};$$

— *sinon, on relève* P *en (x',y) dans* $\mathrm{E}_{n^{k+1}}$, *où*

$$x' \equiv x - \big((3x^2+a)^{-1} \bmod n\big) f(x,y) \pmod{n^{k+1}}.$$

Au bout de $(s-1)$ étapes, on trouve un élément $(\mathrm{P})_l$ *de* $\mathrm{E}_{n^{s+1}}$ *tel que* $(\mathrm{P})_l \equiv \mathrm{P} \pmod n$.

Remarque. Modulo p, on est certain d'être dans l'un des deux cas à chaque étape, sinon le point serait singulier. Comme n est un entier RSA, la méthode fonctionnera avec une probabilité proche de 1.

2.3. Dans les quotients de corps quadratiques

Dans cette sous-section, on adapte le cryptosystème de Paillier dans les quotients de corps quadratiques, à l'aide de la fonction trappe définie en sous-section II–1.2. On obtiendra ainsi un nouveau cryptosystème.

On note toujours $n = pq$ un entier RSA et Δ un entier non carré dans $\mathbf{Z}$ premier avec n. Dans un premier temps, on va travailler avec le groupe $\mathrm{G} := \big(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\big)^\wedge$ défini en sous-section III–3.1, puis on verra une généralisation dans le groupe $\big(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\big)^\wedge$ pour $s > 1$.

Principe

D'après le théorème III – 7 page 52, le groupe $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$ est le produit de deux groupes cycliques $\left(\mathcal{O}_\Delta/p^2\mathcal{O}_\Delta\right)^\wedge$ et $\left(\mathcal{O}_\Delta/q^2\mathcal{O}_\Delta\right)^\wedge$. L'ordre de $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$ est $\varphi_\Delta(n^2) = n\varphi_\Delta(n)$. On va utiliser l'entier $k := n$. On suppose donc que $\mathrm{pgcd}(n, \varphi_\Delta(n)) = 1$ (c'est à dire que $p \nmid (q \pm 1)$ et que $q \nmid (p \pm 1)$). On utilise encore un exposant du groupe :

$$\lambda_\Delta := \mathrm{ppcm}\Big(p - (\Delta/p), q - (\Delta/q)\Big),$$

de telle sorte que

$$\forall x \in \left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge, x^{n\lambda_\Delta} = 1.$$

Comme on a aussi $\mathrm{pgcd}(n, \lambda_\Delta(n)) = 1$, le lemme II – 3 donne

$$\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^{\wedge n} = \left\{x \in (\mathcal{O}_\Delta/(n^2))^\wedge,\, x^{\lambda_\Delta} = 1\right\}.$$

D'après le corollaire II – 2, le groupe quotient $\mathrm{G}/\mathrm{G}^n = \left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge / \left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^{\wedge n}$ est alors d'ordre n.

En fait, comme dans les cas des quotients de $\mathbf{Z}$ et des courbes elliptiques, ce groupe est cyclique puisqu'il existe un élément g d'ordre n dans G. Cet élément est un générateur du noyau de la surjection canonique $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge \longrightarrow \left(\mathcal{O}_\Delta/n\mathcal{O}_\Delta\right)^\wedge$, étudié en sous-section III–3.3. D'après cette étude, on peut prendre $g \equiv 1 + n\sqrt{\Delta} \pmod{n^2}$. De plus, on dispose d'un algorithme immédiat pour calculer le logarithme discret dans g puisque

$$g^m \equiv 1 + mn\sqrt{\Delta} \pmod{n^2},$$

pour tout $m \in \mathbf{Z}/n\mathbf{Z}$.

Chiffrement

Soit $m \in \mathbf{Z}/n\mathbf{Z}$, on chiffre m en

$$c := g^m\beta \equiv (1 + mn\sqrt{\Delta})\beta \pmod{n^2},$$

où β est une puissance n–ième et les calculs sont faits dans $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$.

Déchiffrement

Pour déchiffrer, il suffit de calculer le logarithme discret de c en base g dans le groupe quotient G/G^n. Cela est facile connaissant la trappe λ_Δ. En effet, on a

$$c^{\lambda_\Delta} \equiv (1 + \lambda_\Delta mn\sqrt{\Delta}) \pmod{n^2}.$$

Comme λ_Δ et n sont premiers entre eux, on peut ensuite retrouver m modulo n. Ce procédé peut être optimisé par l'utilisation des restes chinois. Notons que la connaissance de la trappe permet de factoriser n.

Les puissances n–ièmes

Il semble difficile, sans connaître la factorisation de n, de construire des éléments de norme 1 de $\mathcal{O}_\Delta/n\mathcal{O}_\Delta$, Δ étant donné, puisqu'il faut résoudre l'équation $x^2 - \Delta y^2 = 1$ dans $\mathbf{Z}/n^2\mathbf{Z}$. Pour contourner cette difficulté et produire efficacement des puissances n–ièmes on propose deux solutions.

Première solution : Pour générer une puissance n–ième, on peut se restreindre à des Δ non carrés modulo p et q et utiliser la paramétrisation du tore algébrique comme présenté lors de la description de la variante de la fonction RSA quadratique page 77. En effet, comme vu page 22, on a l'isomorphisme

$$(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge \longrightarrow \mathrm{G}^n, \quad \beta \longmapsto \beta^n.$$

Il suffit donc de produire des éléments aléatoires de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ et de les élever à la puissance n. La production d'éléments de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ se fait avec la fonction ψ définie en page 77. La fonction de chiffrement devient alors

$$\mathcal{E}_{(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta)^\wedge,n,g} : \begin{cases} \mathbf{Z}/n\mathbf{Z} \times (\mathbf{Z}/n\mathbf{Z})^\times \longrightarrow \mathrm{G} \\ (m \ , \ r) \longmapsto g^m\psi(r)^n \end{cases}$$

où $\psi(r)^n$ est calculé dans G.

Seconde solution : On utilise la seconde solution donnée en page 23, comme dans le cadre des courbes elliptiques.

On va donc publier une puissance n–ième β. Pour générer d'autres puissances n–ièmes, le chiffreur calcule alors β^r, où r est aléatoire, inférieur à n. La sécurité sémantique est alors réduite à la reconnaissance des éléments de $<\beta>$. Il convient donc de prendre un β d'ordre grand.

Voyons si l'on peut construire une puissance n–ième β d'ordre maximal. Modulo p, on voit facilement, par un raisonnement analogue à celui fait dans la remarque page 20 que le morphisme $x \mapsto x^p$ de $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\wedge$ dans $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\wedge$ donne par passage au quotient l'isomorphisme :

$$(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\wedge \xrightarrow{\sim} (\mathcal{O}_\Delta/p^2\mathcal{O}_\Delta)^{\wedge p}.$$

Comme $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\wedge$ est cyclique, $(\mathcal{O}_\Delta/p^2\mathcal{O}_\Delta)^{\wedge p}$ l'est aussi et on obtient un générateur en considérant ${g_p}^p \pmod{p^2}$ où g_p engendre $(\mathcal{O}_\Delta/p\mathcal{O}_\Delta)^\wedge$.

Par les restes chinois et le fait que p (resp. q) est premier avec l'ordre de $(\mathcal{O}_\Delta/q^2\mathcal{O}_\Delta)^\wedge$ (resp. l'ordre de $(\mathcal{O}_\Delta/p^2\mathcal{O}_\Delta)^\wedge$), on voit facilement que

$$(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta)^{\wedge n} \cong (\mathcal{O}_\Delta/p^2\mathcal{O}_\Delta)^{\wedge p} \times (\mathcal{O}_\Delta/q^2\mathcal{O}_\Delta)^{\wedge q}.$$

On note g_q un générateur de $(\mathcal{O}_\Delta/q\mathcal{O}_\Delta)^\wedge$. Par cet isomorphisme, on peut construire un sous-groupe cyclique de $(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta)^{\wedge n}$ d'ordre λ_Δ en considérant le groupe engendré par la puissance n–ième β correspondant au couple $({g_p}^p, {g_q}^q)$ via l'isomorphisme chinois. On renvoie en sous-section III–3.5 pour voir comment obtenir g_p et g_q.

Au final, si β est d'ordre λ_Δ, la fonction de chiffrement est le morphisme injectif

$$\mathcal{E}_{(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta)^\wedge,n,g,\beta} : \begin{cases} \mathbf{Z}/n\mathbf{Z} \times \mathbf{Z}/\lambda_\Delta\mathbf{Z} & \longrightarrow & \mathrm{G} \\ (m \ , \ r) & \longmapsto & g^m\beta^r \end{cases}$$

En pratique

On utilise les suites de Lucas. Pour le chiffrement, en utilisant, par exemple, la seconde solution pour produire des puissances n–ièmes, on tire r au hasard dans $\{1,\dots,n-1\}$ et on calcule β^r par

$$\beta^r \equiv \frac{\mathrm{V}_r(2\beta_1)}{2} + \beta_2\mathrm{U}_r(2\beta_1,1)\sqrt{\Delta} \pmod{n^2},$$

en notant $\beta \equiv \beta_1 + \beta_2\sqrt{\Delta} \pmod{n^2}$. On utilise pour cela l'algorithme présenté dans la figure III.4, page 56.

Pour le déchiffrement, on utilise les restes chinois. Voyons comment l'on procède. Soit $c \in \left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$ à déchiffrer. On note $c \equiv (1+n\sqrt{\Delta})^m\gamma \pmod{n^2}$, avec $\gamma \in \mathrm{G}^n$. On doit retrouver m dans $\mathbf{Z}/n\mathbf{Z}$. On réduit c modulo p^2 :

$$c \equiv \left(1+[mq \bmod p]p\sqrt{\Delta}\right)\gamma \pmod{p^2}.$$

On doit donc calculer $(m \bmod p)$ qui est le logarithme discret de c en base $1+qp$ dans $\left(\mathcal{O}_\Delta/p^2\mathcal{O}_\Delta\right)^\wedge$ modulo les puissances p–ièmes (les puissances n–ièmes de $\left(\mathcal{O}_\Delta/p^2\mathcal{O}_\Delta\right)^\wedge$ sont ses puissances p–ièmes car $n = pq$ et q est choisi premier avec l'ordre du groupe modulo p). Dans ce groupe, les ordres des puissances p–ièmes divisent $\lambda_\Delta(p) = p - (\Delta/p)$. En élevant c à la puissance $\lambda_\Delta(p)$ dans $\mathcal{O}/(p^2)$, on trouve

$$c^{\lambda_\Delta(p)} \equiv 1 + [\lambda_\Delta(p)mq \bmod p]p\sqrt{\Delta} \pmod{p}^2,$$

puis on récupère la valeur de $m\lambda_\Delta(p)q$ modulo p et enfin celle de m modulo p car $\lambda_\Delta(p)q$ est premier avec p. On procède de même modulo q, puis les restes chinois permettent de retrouver m modulo n.

Efficacité

Côté chiffrement, on estime le coût du calcul des suites de Lucas à $9\,|n|_2$ multiplications modulo n en utilisant l'algorithme de la figure III.4 page 56 avec les estimations de la sous-section III–1. Si on utilise la première solution pour produire des puissances n–ièmes, on rajoute l'évaluation de la fonction ψ, soit trois multiplications et une inversion modulo n.

Pour le déchiffrement, avec les restes chinois, l'étape principale prend autant que $3\,|n|_2$ multiplications modulo n.

L'expansion du cryptosystème est de 4 *a priori*, mais peut être réduite à 3 en stockant un élément γ de $\mathrm{G} = \left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$ sous la forme $(k, \gamma \bmod n) \in \mathbf{Z}/n\mathbf{Z} \times (\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ avec k tel que $\gamma \equiv (1+n\sqrt{\Delta})^k\,\tilde{\gamma} \pmod{n^2}$ où $\tilde{\gamma}$ est le relevé de $(\gamma \bmod n)$ dans $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$ défini par la proposition III – 9, page 58.

Sécurité

La sécurité du système repose sur le calcul du logarithme discret dans le groupe cyclique G/G^n, la factorisation de n étant inconnue. On note ce problème $\text{Classe}_{(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta)^\wedge,n}$. Par le corollaire II – 9, on a la hiérarchie de problèmes :

$$\text{Classe}_{(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta)^\wedge,n} \overset{\mathscr{P}}{\Longleftarrow} \text{RSA}-\text{Q}_{n,n} \overset{\mathscr{P}}{\Longleftarrow} \text{FACT}(n),$$

où le problème $\text{RSA}-\text{Q}_{n,n}$ correspond au problème d'inversion ponctuelle de la fonction RSA quadratique RSA–Q de paramètre (n,n). Cette fonction a été décrite en sous-section 1.2. Si on utilise la première solution pour produire des puissances n–ièmes, on a la même hiérarchie en remplaçant ce problème par celui de l'inversion ponctuelle de la fonction $\text{RSA}-\text{Q}_{\Delta,n,n}$ décrite page 77.

On retrouve donc une situation similaire à celle que l'on a vue dans les quotients de $\mathbf{Z}$ et les courbes elliptiques.

En utilisant la première solution, la sécurité sémantique est basée sur la reconnaissance des éléments de $\psi\left((\mathbf{Z}/n\mathbf{Z})^\times\right)$ dans $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$. Remarquons que les éléments de G^n qui ne sont pas dans $\psi\left((\mathbf{Z}/n\mathbf{Z})^\times\right)$ sont modulo p ou modulo q congrus à ± 1. Parmi ces éléments, 1 et -1 sont facilement reconnaissables et les autres permettent de factoriser n. La reconnaissance des éléments de $\psi\left((\mathbf{Z}/n\mathbf{Z})^\times\right)$ est donc équivalente à celle des éléments de G^n.

Si on utilise la deuxième solution, la sécurité sémantique est basée sur la reconnaissance des éléments du sous-groupe engendré par β.

Généralisation dans $\left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^\wedge$ avec $s > 1$

On va travailler dans $G := \left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^\wedge$ de discriminant Δ, avec $s > 1$. Par le théorème III – 7, ce groupe est d'ordre $n^s\varphi_\Delta(n)$. On suppose que $\text{pgcd}(n, \varphi_\Delta(n)) = 1$ et on note toujours $\lambda_\Delta = \text{ppcm}(p-(\Delta/p), q-(\Delta/q))$, de telle sorte que, d'après le lemme II – 3,

$$\left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^{\wedge n^s} = \left\{x \in \left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^\wedge,\, x^{\lambda_\Delta} = 1\right\}.$$

On pose $k := n^s$. D'après le corollaire II – 2, le groupe quotient G/G^{n^s} est d'ordre n^s. Comme dans le cas $s = 1$, ce groupe est cyclique et on obtient un générateur en considérant un élément α engendrant le noyau de la surjection canonique $\left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^\wedge \to \left(\mathcal{O}_\Delta/n\mathcal{O}_\Delta\right)^\wedge$. On pose $g := \alpha$ où α est donné par l'expression III.18 établie en sous-section III–3.3. Le calcul du logarithme discret en base g dans $\langle g\rangle$ est plus complexe que dans le cas $s = 1$ mais peut toujours se faire avec un coût faible grâce à l'algorithme donné figure III.5, page 61.

En ce qui concerne la génération d'une puissance n^s–ième, pour éviter de devoir faire une exponentiation coûteuse à la puissance n^s, on privilégie la seconde solution exposée dans le cas $s = 1$, *i.e.*, on publie une puissance n^s–ième, β, d'ordre grand (on peut toujours en obtenir une d'ordre maximal λ_Δ en considérant des générateurs de $\left(\mathcal{O}_\Delta/p\mathcal{O}_\Delta\right)^\wedge$ et de $\left(\mathcal{O}_\Delta/q\mathcal{O}_\Delta\right)^\wedge$).

La fonction de chiffrement est alors le morphisme injectif

$$\mathscr{E}_{(\mathscr{O}_\Delta/n^{s+1}\mathscr{O}_\Delta)^\wedge,n^s,g,\beta} : \begin{cases} \mathbf{Z}/n^s\mathbf{Z} \times \mathbf{Z}/\lambda_\Delta\mathbf{Z} \longrightarrow \mathrm{G} \\ (m \ , \ r) \longmapsto g^m\beta^r \end{cases}$$

En pratique, comme λ_Δ est secret, les aléas sont tirés au hasard entre 1 et n.

Le déchiffrement utilise la trappe λ_Δ, comme dans le cas $s = 1$. La sécurité du système et sa sécurité sémantique s'étudient aussi de manière similaire.

L'expansion du système est de $1+2/s$ en utilisant la méthode de « stockage » des points de G évoquée dans le cas $s = 1$. Concernant le coût des exponentiations utilisées par le cryptosystème, pour le chiffrement on trouve

$$\frac{3}{2}(s+1)(s+2)|n|_2 \,\mathrm{M}$$

et pour le déchiffrement

$$\frac{1}{2}(s+1)(s+2)|n|_2 \,\mathrm{M}$$

où M désigne la multiplication modulo n, en utilisant les estimations habituelles. Notons que comme dans les quotients de $\mathbf{Z}$, utiliser une fois le système pour chiffrer un bloc de $s|n|_2$ bits avec $s > 1$ ne sera pas plus rapide que d'appliquer s fois le chiffrement du système dans le cas $s = 1$.

2.4. Comparaison des systèmes

On résume dans le tableau les caractéristiques de trois des systèmes probabilistes homomorphiques présentés précédemment. Ces trois systèmes sont le système de Paillier, son adaptation dans les courbes elliptiques faite par Galbraith et celle dans les corps quadratiques exposée précédemment. Pour le coût des cryptosystèmes, on compte toutes les opérations et non pas seulement les plus coûteuses comme fait dans le reste de la section. Pour le système de Paillier quadratique, on utilise la seconde solution pour produire des puissances n–ièmes.

Cryptosystème	Paillier	Galbraith	Paillier quadratique
Cadre	$(\mathbf{Z}/n^2\mathbf{Z})^\times$	E_{n^2}	$(\mathscr{O}_\Delta/n^2\mathscr{O}_\Delta)^\wedge$
Taille de l'entrée	$\|n\|_2$		
Taille de la sortie	$2\|n\|_2$	$3\|n\|_2$	
Expansion	2	3	
Coût du chiffrement	$(\frac{9}{2}\|n\|+1)\mathrm{M}$	$(35\|n\|+3)\mathrm{M}$	$(9\|n\|+7)\mathrm{M}$
Coût de déchiffrement	$(\frac{3}{2}\|n\|+\frac{5}{3})\mathrm{M}$	$(21\|n\|+\frac{5}{3})\mathrm{M}$	$(3\|n\|+\frac{4}{3})\mathrm{M}+2\mathrm{L}$

Dans le tableau, on a utilisé les estimations de habituelles (cf. page 88). Pour les inversions intervenant dans les formules d'additions dans les courbes elliptiques, une inversion

modulo n a été jugée aussi coûteuse que 10 inversions modulo n. On désigne par L le temps de calcul d'un symbole de Legendre dans $\mathbf{Z}/p\mathbf{Z}$ avec $|p|_2 = |n|_2/2$. Ce calcul est de complexité $\mathcal{O}(|p|_2^2)$.

On retrouve la situation vue pour les systèmes déterministes. Le système présenté dans les corps quadratiques est beaucoup plus intéressant en terme de complexité que ceux proposés dans les courbes elliptiques.

3. Cryptosystèmes probabilistes non homomorphiques

Dans cette section, on va utiliser les fonctions trappe définies en section II– 2 pour produire des systèmes probabilistes. Ces systèmes auront une meilleure complexité que ceux vus dans la section précédente, au prix de la perte de l'homomorphie.

Dans la première sous-section, on va utiliser les quotients de $\mathbf{Z}$, ensuite on utilisera les courbes elliptiques et enfin les corps quadratiques. Avec la première fonction trappe définie en sous-section II–2.1, on adaptera en fait les idées de [CGHGN01]. La deuxième fonction trappe définie en sous-section II–2.2 donnera un système complètement original.

On présentera dans la dernière sous-section un tableau récapitulant les propriétés des principaux cryptosystèmes décrits.

3.1. Dans les quotients de Z

Le système de Catalano, Gennaro *et al.* (2001)

Pour accélérer le chiffrement du système de Paillier, présenté page 84, Catalano, Gennaro *et al.*, dans [CGHGN01], décident d'utiliser au lieu de l'exposant n, un e petit tel que $\mathrm{pgcd}\big(e, \varphi(n)\big) = 1$, comme dans RSA. C'est cette démarche que l'on a généralisée en sous-section II–2.1. On va décrire le cryptosystème que l'on obtient en utilisant les résultats de II–2.1. On utilise le paragraphe « Construction de la fonction f » de cette sous-section.

Cadre : On se place dans le groupe $\mathrm{G} := \left(\mathbf{Z}/n^2\mathbf{Z}\right)^\times$. On note $\mathrm{H} := (\mathbf{Z}/n\mathbf{Z})^\times$ de telle sorte que l'élément $g \equiv 1 + n \pmod{n^2}$ soit un générateur du noyau (d'ordre n) de la surjection canonique de G sur H.

On note respectivement Ω et Λ les ensembles sous-jacents de G et H, *i.e.*,

$$\Omega := \left\{ r \in \mathbf{N}, 0 < r < n^2, \mathrm{pgcd}(r, n) = 1 \right\},$$

et

$$\Lambda := \left\{ r \in \mathbf{N}, 0 < r < n, \mathrm{pgcd}(r, n) = 1 \right\}.$$

Avec les notations de II–2.1, on a en fait $\overline{\Lambda} := \Lambda$. L'ensemble Λ est un système de représentants des classes de Ω modulo n.

Soit e premier avec $\varphi(n)$, l'automorphisme RSA, $x \mapsto x^e$, de H induit la permutation $\overline{f} : x \mapsto (x^e \bmod n)$ de Λ. On relève cette fonction de Λ dans Ω en la fonction

$$f : x \longmapsto (x^e \bmod n^2).$$

On est alors dans les hypothèses de la sous-section II–2.1.

Fonction de chiffrement : On considère la fonction de chiffrement suivante.

$$\mathscr{E}_{(\mathbf{Z}/n^2\mathbf{Z})^\times,\mathrm{RSA}_{n,e},g} : \left\{ \begin{array}{ccccc} \mathbf{Z}/n\mathbf{Z} & \times & \Lambda & \longrightarrow & \Omega \\ (m & , & r) & \longmapsto & g^m r^e \bmod n^2 \end{array} \right.$$

où m est le message clair et r un aléa. Par construction, cette fonction est bien une bijection puisque $n|\Lambda| = |\Omega|$ et car la fonction $x \mapsto \big(f(x) \bmod n\big)$ de Λ dans Λ est bien injective (elle est même bijective).

Clef privée et déchiffrement : La clef privée du cryptosystème est la trappe permettant d'inverser $\overline{f}$, c'est donc d l'inverse de e modulo $\lambda(n)$.

Pour déchiffrer un élément de c de Ω, on le réduit modulo n, puis on inverse la fonction $\overline{f}$, *i.e.*, on calcule $r := c^d \bmod n$. On relève ce résultat modulo n^2. Ici c'est immédiat, puis on récupère g^m en calculant $c/f(r)$ dans G. On en déduit m par le calcul direct du logarithme discret dans $\langle g \rangle$, comme on l'a vu avec le système de Paillier, page 84.

Sécurité : La sécurité du cryptosystème est basée sur le problème suivant : étant donné c un élément de Ω, trouver m dans $\mathbf{Z}/n\mathbf{Z}$ tel qu'il existe r dans Λ tel que $c = g^m f(r)$. On note $\mathrm{Classe}_{(\mathbf{Z}/n^2\mathbf{Z})^\times,\mathrm{RSA}_{n,e},g}$ ce problème. Par le déchiffrement, on a

$$\mathrm{Classe}_{(\mathbf{Z}/n^2\mathbf{Z})^\times,\mathrm{RSA}_{n,e},g} \overset{\mathscr{P}}{\Longleftarrow} \mathrm{RSA}_{n,e}.$$

On a en fait l'équivalence, d'après les théorèmes II – 14 et II – 15 :

$$\mathrm{Classe}_{(\mathbf{Z}/n^2\mathbf{Z})^\times,\mathrm{RSA}_{n,e},g} \overset{\mathscr{P}}{\Longleftrightarrow} \mathrm{RSA}_{n,e}, \tag{IV.3}$$

c'est le résultat obtenu dans [CNS02].

D'après le théorème II – 17, la sécurité sémantique est équivalente à la difficulté du problème de reconnaissance des éléments de $f(\Lambda)$ dans Ω.

Efficacité : Le coût du chiffrement est essentiellement celui d'une exponentiation à la puissance e (petit) modulo n^2. En moyenne sur les entiers e utilisés ceci se fait en $9|e|_2/2$ multiplications modulo n, mais des entiers e avec un poids de Hamming faible permettent d'optimiser ce coût.

Pour le déchiffrement, on a essentiellement besoin d'une exponentiation modulo n à la puissance d (avec d inférieur à $\lambda(n)$) puis d'une exponentiation à la puissance e. Avec les restes chinois, on obtient $|n|_2/2 + 3|e|_2$ multiplications modulo n en moyenne.

Comparé au système de Paillier, on gagne sur le chiffrement comme voulu, le déchiffrement reste lui de complexité comparable. L'expansion du système reste de 2.

On obtient ainsi un système compétitif qui peut être décrit comme un système RSA probabiliste. De plus, la sécurité est bien maîtrisée car on a une équivalence avec celle du système RSA.

Généralisation : Tout comme le système de Paillier, le système précèdent peut être généralisé en prenant $G := \left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times$ avec $s > 1$, afin de diminuer l'expansion. On décrit brièvement cette généralisation.

On a toujours $H = (\mathbf{Z}/n\mathbf{Z})^\times$ et $g \equiv 1 + n \pmod{n}^{s+1}$. L'ensemble Λ reste inchangé mais Ω est maintenant défini par

$$\Omega := \left\{ r \in \mathbf{N},\, r < n^{s+1}, \mathrm{pgcd}(r,n) = 1 \right\}.$$

On garde la même fonction trappe $\overline{f}$, permutation de Λ, mais on la relève maintenant de Λ dans Ω en la fonction

$$f : x \longmapsto (x^e \bmod n^{s+1}).$$

La fonction de chiffrement devient la bijection suivante

$$\mathcal{E}_{\left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times, \mathrm{RSA}_{n,e}, g} : \left\{ \begin{array}{ccccc} \mathbf{Z}/n^s\mathbf{Z} & \times & \Lambda & \longrightarrow & \Omega \\ (m & , & r) & \longmapsto & g^m r^e \bmod n^{s+1} \end{array} \right.$$

Le déchiffrement se fait comme dans le cas $s = 1$.

On note $\mathrm{Classe}_{\left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times, \mathrm{RSA}_{n,e}, g}$ le problème sur lequel repose la sécurité du système. Par construction, on a toujours

$$\mathrm{Classe}_{\left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times, \mathrm{RSA}_{n,e}, g} \overset{\mathscr{P}}{\Longleftarrow} \mathrm{RSA}_{n,e}.$$

Comme dans le cadre des systèmes de Damgård-Jurik et de Paillier (cf. (IV.2)), on montre facilement que

$$\mathrm{Classe}_{\left(\mathbf{Z}/n^{2}\mathbf{Z}\right)^\times, \mathrm{RSA}_{n,e}, g} \overset{\mathscr{P}}{\Longleftarrow} \mathrm{Classe}_{\left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times, \mathrm{RSA}_{n,e}, g}.$$

Avec (IV.3), on obtient que

$$\mathrm{Classe}_{\left(\mathbf{Z}/n^{2}\mathbf{Z}\right)^\times, \mathrm{RSA}_{n,e}, g} \overset{\mathscr{P}}{\Longleftrightarrow} \mathrm{Classe}_{\left(\mathbf{Z}/n^{s+1}\mathbf{Z}\right)^\times, \mathrm{RSA}_{n,e}, g} \overset{\mathscr{P}}{\Longleftrightarrow} \mathrm{RSA}_{n,e}.$$

De même, le résultat sur le sécurité sémantique obtenu par Damgård et Jurik dans [DJ01] s'adapte ici et on obtient que la sécurité sémantique du système avec $s > 1$ est équivalente à celle du système de Catalano *et al.*

On obtient donc un système de sécurité équivalente, adapté à des situations différentes suivant la taille des blocs à chiffrer et suivant l'expansion recherchée (elle est ici de $1 + 1/s$).

Le coût augmente avec s, plus précisément, l'exponentiation du chiffrement coûte en moyenne autant que $3(s+1)(s+2)|e|_2/2$ multiplications modulo n et celle du déchiffrement, autant que $|n|_2/2 + |e|_2(s+1)(s+2)/2$ multiplications modulo n. Ainsi, comme l'étape la plus importante du déchiffrement reste un déchiffrement RSA modulo n qui ne dépend pas de s, le coût augmente de façon relativement faible quand s augmente.

Un cryptosystème basé sur la fonction trappe de II–2.2

Dans cette section, on utilise la fonction trappe définie en sous-section II–2.2 pour proposer un système probabiliste non homomorphique dans $G := (\mathbf{Z}/n\mathbf{Z})^\times$. Pour cela, on a besoin de deux fonctions trappe déterministes, f et h permutations de l'ensemble sous-jacent de G.

Cadre : Soit e un entier premier avec $\varphi(n)$, on peut déjà utiliser la fonction $\mathrm{RSA}_{n,e}$, c'est à dire poser $f : \mathrm{G} \to \mathrm{G}, x \mapsto x^e$.

Comme la fonction $\mathrm{RSA}_{n,e}$ est un morphisme de G, on doit utiliser pour h une fonction différente sinon le système ne sera pas sémantiquement sûr (cf. la remarque de la page 31). On va utiliser la fonction trappe introduite en sous-section 1.2. On pose $h = \mathrm{LUC}_{n,e}$, en supposant de plus que e est premier avec $(p^2-1)(q^2-1)$.

D'après le corollaire IV – 5, la fonction h est une permutation de

$$\left\{x \in \mathbf{N},\, x < n,\, \gcd(x^2-4, n) = \gcd(x, n) = 1\right\}.$$

Cependant, on voit facilement, par définition de la suite de Lucas V (cf. page 53), que pour tout entier k, $\mathrm{V}_k(2) = 2$ et $\mathrm{V}_k(-2) = -2$. Ainsi h induit une permutation de $(\mathbf{Z}/p\mathbf{Z})^\times$ et de $(\mathbf{Z}/q\mathbf{Z})^\times$. Par les restes chinois, h est donc bien une permutation de l'ensemble sous-jacent de G.

Chiffrement : La fonction de chiffrement est la bijection :

$$\mathcal{E}_{\mathrm{RSA}_{n,e},\mathrm{LUC}_{n,e}} : \left\{ \begin{array}{ccccccc} (\mathbf{Z}/n\mathbf{Z})^\times & \times & (\mathbf{Z}/n\mathbf{Z})^\times & \longrightarrow & (\mathbf{Z}/n\mathbf{Z})^\times & \times & (\mathbf{Z}/n\mathbf{Z})^\times \\ (m & , & r) & \longmapsto & \left(\mathrm{V}_e(mr) & , & r^{-e}\right) \end{array} \right.$$

où $\mathrm{V}_e(m, r)$ désigne, par abus de notation, l'élément c de G tel que $c \equiv \mathrm{V}_e(m, r) \pmod{n}$.

Déchiffrement : Pour déchiffrer on inverse les deux fonctions. Cela peut se faire brutalement avec un inverse d de e modulo $(p^2-1)(q^2-1)$, ou plus finement, en utilisant les restes chinois et des inverses de e modulo $p \pm 1$ et modulo $q \pm 1$, comme vu en sous-section 1.2. On récupère le message en multipliant dans G les deux résultats obtenus.

Sécurité : D'après l'analyse faite en sous-section II–2.2, le problème décrivant la sécurité du système est équivalent au problème de l'inversion ponctuelle des deux fonctions, RSA et LUC. Pour la sécurité sémantique, le système sera sûr s'il n'existe pas d'adversaire capable d'exhiber un élément m de G pour lequel il sera reconnaître dans G^2 les éléments de la forme (c_1, c_2) tel que $v_d(c_1)c_2^{-d} \equiv m \bmod n$.

Efficacité : L'expansion du système est de deux. Le coût principal du chiffrement, celui de l'évaluation de f et g sera en moyenne de $7\,|e|_2\,/2$ multiplications modulo n et celui du déchiffrement de $7\,|n|_2\,/6$ multiplications modulo n.

On obtient donc un système très simple et très compétitif en chiffrement, comme on le verra en sous-section 3.4. Cependant, la sécurité sémantique repose sur un problème qu'on ne sait pas analyser.

3.2. Dans les courbes elliptiques, le système de Galindo *et al.*

Dans cette sous-section, on expose le système de Galindo, Martin *et al.* présenté dans [GMMV02]. Ce système peut être vu de diverses manières : comme une adaptation du

système de Catalano *et al.* dans les courbes elliptiques, comme une variante plus rapide mais non homomorphique du système de Galbraith (cf. sous-section 2.2), ou enfin comme un KMOV probabiliste (cf. sous-section 1.1).

On va exposer ce système à l'aide de la fonction trappe de la sous-section II–2.1. Comme on va travailler avec des groupes du type E_{n^2}, il sera difficile pour celui qui ne connaît pas la factorisation de n d'exhiber des éléments aléatoires du groupe. On va donc adapter quelque peu cette fonction trappe pour pouvoir utiliser des groupes variables.

Relèvement des éléments

Le principal changement venant de l'utilisation de groupes variables va être, en utilisant les notations du paragraphe « Construction de la fonction f » (cf. page 26), au niveau du relèvement des éléments de $\overline{\Lambda}$ dans Λ. Voyons cela plus en détail.

Soit $\mathrm{M} \coloneqq (x, y)$ un couple de $\mathbf{Z}/n^2\mathbf{Z} \times \mathbf{Z}/n^2\mathbf{Z}$. On suppose que $b \coloneqq y^2 - x^3$ est inversible de telle sorte que M soit un point de la courbe $\mathrm{E}_{n^2}(0, b)$. On a vu (cf. III–2.5) qu'il existe une surjection de $\mathrm{E}_n^2(0, b)$ sur $\mathrm{E}_n\big(0, b \pmod n\big)$ de noyau E_1 dont l'expression ne dépend pas de b. On rappelle que E_1 est d'ordre n. De plus, comme vu dans l'exposition de système de Galbraith (cf. sous-section 2.2), ce groupe est cyclique d'ordre n, engendré par l'élément $g \coloneqq \mathcal{O}_1 \equiv (n : 1 : 0)$.

Étant donnés un couple d'entiers $(\tilde{x}, \tilde{y})$ tel que $(\tilde{x}, \tilde{y}) \equiv \Pi(\mathrm{M}) \pmod n$ et b caractérisant la courbe $\mathrm{E}_{n^2}(0, b)$, on veut pouvoir retrouver M parmi les éléments de $\mathrm{M} + \mathrm{E}_1$. Il faut donc que l'ensemble dans lequel est choisi le couple (x, y) de départ soit tel qu'il n'y ait qu'un relèvement possible du point $\Pi(\mathrm{M})$ de la courbe modulo n dans la courbe $\mathrm{E}_{n^2}(0, b)$ ayant ses coordonnées dans cet ensemble.

Si $\tilde{y}$ est premier avec n, par le lemme de Hensel, il existe un unique relevé de $\Pi(\mathrm{M})$ congru à $(\tilde{x}, y')$ où y' est un élément bien déterminé modulo n^2. Cet élément est donné dans la proposition IV – 6 :

$$y' \equiv \tilde{y} - \big((2\tilde{y})^{-1} \bmod n\big)\big(\tilde{y}^2 - \tilde{x}^3 - b\big) \pmod{n^2}.$$

On note alors

$$\Lambda \coloneqq \Big\{(x, y) \in \mathbf{N}^2,\ x < n,\ y < n^2,\ \mathrm{pgcd}(y, n) = \mathrm{pgcd}(y^2 - x^3, n) = 1\Big\}.$$

Ainsi, si (x, y) est un couple d'entiers appartenant à Λ et si b est l'élément de $\big(\mathbf{Z}/n^2\mathbf{Z}\big)^\times$ tel que $b \equiv y^2 - x^3 \pmod{n^2}$, alors le point M congru à (x, y) modulo n^2 de la courbe $\mathrm{E}_{n^2}(0, b)$ est tel qu'étant donnés $\Pi(\mathrm{M})$ et b, il existe un seul et unique relèvement de $\Pi(\mathrm{M})$ dans la courbe $\mathrm{E}_{n^2}(0, b)$ dont les coordonnées sont congrues modulo n^2 à un élément de Λ. Cet élément est M.

Muni de cette méthode pour relever les éléments, on va construire le cryptosystème à l'aide de la fonction KMOV présentée en sous-section 1.1 en s'inspirant de la méthode donnée page 26.

Cadre

On suppose que $n = pq$ avec $p \equiv q \equiv 2 \pmod 3$ et que e est un entier premier avec $(p+1)(q+1)$. On note

$$\overline{\Lambda} := \left\{(x,y) \in \mathbf{N}^2,\, x < n,\, y < n,\, \mathrm{pgcd}(y,n) = \mathrm{pgcd}(y^2 - x^3, n) = 1\right\}.$$

D'après la proposition IV – 2, la fonction $\overline{f}$ qui à un couple (x,y) de $\overline{\Lambda}$ associe l'élément $\left(\mathrm{KMOV}_{n,e}(\mathrm{M}) \bmod n\right)$, où $\mathrm{M} \equiv (x,y) \pmod n$, est une permutation de $\overline{\Lambda}$.

On va vouloir relever la fonction $\overline{f}$ modulo n^2 de manière à pouvoir inverser la fonction obtenue en se ramenant modulo n. On pose

$$\Omega := \left\{(x,y) \in \mathbf{N}^2,\, x < n^2,\, y < n^2,\, \mathrm{pgcd}(y,n) = \mathrm{pgcd}(y^2 - x^3, n) = 1\right\}.$$

Le relèvement utilisé sera la fonction f de Λ dans Ω

$$f : (x,y) \longmapsto e.\mathrm{M} \bmod n^2,$$

où M désigne l'élément de $\mathbf{Z}/n^2\mathbf{Z} \times \left(\mathbf{Z}/n^2\mathbf{Z}\right)^\times$ tel que $\mathrm{M} \equiv (x,y) \pmod{n^2}$. C'est un point de la courbe $\mathrm{E}_{n^2}(0,b)$ avec $b \equiv y^2 - x^3 \pmod{n^2}$. Le calcul de l'exponentiation est effectué dans cette courbe.

Fonction de chiffrement

La fonction de chiffrement utilisée par Galindo *et al.* est

$$\mathscr{E}_{\mathrm{E}_{n^2},\mathrm{KMOV}_{n,e},\mathscr{O}_1} : \left\{ \begin{array}{ccccc} \mathbf{Z}/n\mathbf{Z} & \times & \Lambda & \longrightarrow & \Omega \\ (m & , & (x,y)) & \longmapsto & \left(m.\mathscr{O}_1 + e.\mathrm{P}\right) \bmod n^2 \end{array} \right.$$

où $\mathrm{P} \equiv (x,y) \pmod{n^2}$ est un point de la courbe $\mathrm{E}_{n^2}(0,b)$ avec $b \equiv y^2 - x^3 \pmod{n^2}$ et les calculs de la parenthèse sont effectués dans cette même courbe.

Cette fonction est bien définie par construction. De plus, c'est une bijection. En effet, on a $n\,|\Lambda| = |\Omega|$ et la fonction est injective toujours par construction.

Clef privée et déchiffrement

La clef secrète de déchiffrement est l'inverse d de e modulo $\mathrm{ppcm}(p+1, q+1)$. Soit $\mathrm{C} := (x,y) \in \Omega$ un chiffré de $m \in \mathbf{Z}/n\mathbf{Z}$ avec l'aléa (x_r, y_r) de Λ. Voyons comment déchiffrer C. On commence par réduire C modulo n. On obtient un élément de $\overline{\Lambda}$ que l'on inverse par la fonction $\overline{f}$. Ainsi, on voit le couple réduit comme un élément Q de la courbe $\mathrm{E}_n(0,\tilde{b})$ où $\tilde{b}$ est l'élément de $(\mathbf{Z}/n\mathbf{Z})^\times$ tel que $y^2 - x^3 \equiv b \pmod n$ et on inverse cet élément par la fonction $\mathrm{KMOV}_{n,e}$, *i.e.*, on calcule $\mathrm{P} := d.\mathrm{Q}$ dans $\mathrm{E}_n(0,\tilde{b})$.

Soit b l'élément de $\left(\mathbf{Z}/n^2\mathbf{Z}\right)^\times$ tel que $y^2 - x^3 \equiv b \pmod{n^2}$, on relève P de la manière exposée dans le paragraphe « relèvement des éléments », afin d'obtenir un point de la courbe $\mathrm{E}_{n^2}(0,b)$ dont les coordonnées sont congrues à un élément de Λ. Par construction on retrouve alors l'aléa (x_r, y_r). Il ne reste plus qu'à retrouver $m.\mathscr{O}_1$ en voyant C et (x_r, y_r) comme des points de $\mathrm{E}_{n^2}(0,b)$: $m.\mathscr{O}_1 = \mathrm{C} - e.(x_r, y_r)$.

Sécurité

Le problème du déchiffrement se réduit donc à celui du système KMOV de paramètres n et e. On peut toujours exprimer la sécurité avec le problème Hensel (cf. théorème II – 14). Par contre le résultat de Catalano *et al.* (cf. théorème II – 15) ne s'applique pas, l'équivalence du système de Galindo et du système KMOV n'est pas connue.

Comme la fonction f est basée sur la fonction KMOV, la même analyse que celle faite en 1.3 sur la taille du paramètre e s'applique ici. Ainsi, Galindo *et al.* (cf. section 4.2 de [GMMV02]) estiment que leur système avec exposant e a le même niveau de sécurité que celui de Catalano avec un exposant e^2.

Même si on utilise des groupes variables, le théorème II – 17 s'applique encore ici (le système vérifie toujours la propriété énoncée dans la preuve du théorème). La sécurité sémantique du schéma repose donc sur la difficulté de la reconnaissance des éléments de $f(\Lambda)$ dans Ω.

Efficacité

Le coût du chiffrement est essentiellement celui de l'exponentiation eP dans E_{n^2}, car le calcul de $m.\mathscr{O}_1$ est direct comme vu en sous-section 2.2. D'après les estimations de cette sous-section, ce calcul se fait en moyenne sur les entiers e utilisés en près de

$$20\,|e|_2\,\mathrm{M} + \frac{3}{2}\,|e|_2\,\mathrm{I}.$$

On verra en sous-section 3.4 que comme annoncé par les auteurs de [GMMV02], le système chiffre plus rapidement que le système El Gamal elliptique.

Pour le déchiffrement, en utilisant les restes chinois, on trouve près de

$$\left(\frac{3}{2}\,|n|_2 + 20\,|e|_2\right)\mathrm{M} + \left(\frac{1}{6}\,|n|_2 + \frac{3}{2}\,|e|_2\right)\mathrm{I}.$$

Généralisation dans $\mathrm{E}_{n^{s+1}}$, avec $s > 1$

Pour améliorer l'expansion, on peut toujours augmenter s. Voyons cela dans les grandes lignes. On redéfinit les ensembles Λ et Ω de la manière suivante :

$$\Lambda := \left\{(x,y) \in \mathbf{N}^2,\, x < n,\, y < n^{s+1},\, \mathrm{pgcd}(y,n) = \mathrm{pgcd}(y^2 - x^3, n) = 1\right\},$$

$$\Omega := \left\{(x,y) \in \mathbf{N}^2,\, x < n^{s+1},\, y < n^{s+1},\, \mathrm{pgcd}(y,n) = \mathrm{pgcd}(y^2 - x^3, n) = 1\right\}.$$

On utilise alors la fonction de chiffrement :

$$\mathscr{E}_{\mathrm{E}_{n^{s+1}},\mathrm{KMOV}_{n,e},\mathscr{O}_1} : \left\{ \begin{array}{ccccc} \mathbf{Z}/n^s\mathbf{Z} & \times & \Lambda & \longrightarrow & \Omega \\ (m & , & (x,y)) & \longmapsto & (m.\mathscr{O}_1 + e.\mathrm{P}) \bmod n^{s+1} \end{array} \right.$$

où $\mathcal{O}_m := m\big(n : 1 : w(n)\big)$ et $\mathrm{P} \equiv (x, y) \pmod{n^{s+1}}$ est un point de la courbe $\mathrm{E}_{n^{s+1}}(0, b)$ avec $b \equiv y^2 - x^3 \pmod{n^{s+1}}$ et les calculs de la parenthèse sont effectués dans cette même courbe.

On voit toujours facilement que $\mathscr{E}$ est une bijection. Pour le déchiffrement, comparé à la situation $s = 1$, un point change : le relèvement d'un point de la courbe réduite modulo n en un point de la courbe modulo n^{s+1} ayant ses coordonnées congrues à un élément de Λ. Ce relèvement se fait par récurrence comme vu dans la définition IV – 6. Le calcul final du logarithme discret en base $\mathcal{O}_1$ est lui aussi plus compliqué. Il se fait comme vu dans la sous-section 2.2.

L'analyse de la sécurité est similaire à celle faite dans le cas $s = 1$.

3.3. Dans les quotients de corps quadratiques

Dans cette section, on va adapter les idées précédentes dans les corps quadratiques. On va encore se servir de la construction exposée page 26 en partant de deux fonctions déterministes : la fonction RSA quadratique et la fonction LUC décrite en sous-section 1.2.

Une première adaptation

On va dans un premier temps se servir de la fonction RSA quadratique. Le système en résultant pourra être vu comme une variante rapide mais non homomorphique du système exposé en sous-section 2.3. La construction est très proche de celle vue dans les courbes elliptiques. En effet, on va encore utiliser des groupes variables (comme le système que l'on va produire sera de toutes façons non homomorphique, il est inutile ici d'utiliser la variante de la fonction RSA quadratique utilisant un discriminant fixe exposée page 77).

Relèvement des éléments : Pour se ramener dans le cadre du paragraphe « Construction de la fonction f », page 26, on définit, comme dans le cas du système de Galindo *et al.*, un ensemble qui permettra d'avoir un relèvement unique des éléments de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ dans $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$.

On pose

$$\Lambda := \Big\{(x, y) \in \mathbf{N}^2,\ x < n,\ y < n^2,\ \mathrm{pgcd}(x^2 - 1, n) = \mathrm{pgcd}(y, n) = 1\Big\}.$$

Soit (x, y) un élément de Λ et Δ un entier tel que $\Delta \equiv (x^2 - 1)y^{-2} \pmod{n^2}$. On note ρ un entier quadratique tel que $\rho \equiv x + y\sqrt{\Delta} \pmod{n^2}$. Modulo $n^2\mathcal{O}_\Delta$, ρ est alors un élément de $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$. On note Π la surjection canonique de $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$ sur $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$. On a vu que le noyau de Π est cyclique d'ordre n, engendré par $g \equiv 1 + n\sqrt{\Delta} \pmod{n^2}$.

Étant donnés $\Pi(\rho)$ et Δ, il existe un seul et unique relèvement de $\Pi(\rho)$, de $(\mathcal{O}_\Delta/n\mathcal{O}_\Delta)^\wedge$ dans $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$, congru modulo n^2 à un élément de la forme $x + y'\sqrt{\Delta}$ où y' est un entier donné par la proposition III – 9. On a donc $y' \equiv y \pmod{n^2}$.

Cadre : Soit e un entier premier avec $(p^2-1)(q^2-1)$. D'après la proposition IV – 3, la fonction $\overline{f} := \text{RSA–Q}_{n,e}$ est un permutation de l'ensemble

$$\overline{\Lambda} := \left\{(x,y) \in \mathbf{N}^2,\, x < n,\, y < n,\, \text{pgcd}(x^2-1,n) = \text{pgcd}(y,n) = 1\right\}.$$

On pose

$$\Omega := \left\{(x,y) \in \mathbf{N}^2,\, x < n^2,\, y < n^2,\, \text{pgcd}(x^2-1,n) = \text{pgcd}(y,n) = 1\right\},$$

et on relève la fonction $\overline{f}$ en une fonction f de Λ dans Ω :

$$f : (x,y) \longmapsto (c_1,c_2),$$

où (c_1,c_2) est tel que

$$c_1 + c_2\sqrt{\Delta} \;\equiv\; \left(m_1 + m_2\sqrt{\Delta}\right)^e \pmod{n^2\mathcal{O}_\Delta},$$

l'exponentiation à la puissance e étant faite dans le groupe $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$ avec Δ un entier non carré tel que $\Delta \equiv (m_1^2-1)m_2^{-2} \pmod{n}$.

Fonction de chiffrement : On utilise la fonction suivante :

$$\mathcal{E}_{\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge,\text{RSA–Q}_{n,e},g} : \left\{\begin{array}{ccccc} \mathbf{Z}/n\mathbf{Z} & \times & \Lambda & \longrightarrow & \Omega \\ (m & , & (x,y)) & \longmapsto & (c_1,c_2) \end{array}\right.$$

où (c_1,c_2) est l'élément de Ω tel que

$$c_1 + c_2\sqrt{\Delta} \;\equiv\; g^m\rho^e \pmod{n^2\mathcal{O}_\Delta},$$

avec $\Delta \equiv (x^2-1)y^{-2} \pmod{n^2}$, $\rho \equiv x + y\sqrt{\Delta} \pmod{n^2\mathcal{O}_\Delta}$ un élément de $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$ et g un autre élément de $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$ congru à l'entier quadratique $1 + n\sqrt{\Delta}$.

Cette fonction de chiffrement est bien définie et bijective par construction.

Clef privée et déchiffrement : La clef secrète de déchiffrement est celle permettant d'inverser la fonction $\text{RSA–Q}_{n,e}$, soit, l'inverse d de e modulo $(p^2-1)(q^2-1)$ ou des valeurs plus fines si on utilise les restes chinois (cf. page 73).

Le déchiffrement est similaire à celui du système exposé précédemment dans les courbes elliptiques. On réduit le chiffré modulo n et on l'inverse par la fonction $\overline{f}$. On relève le résultat comme expliqué plus haut puis on évalue la fonction f. On en déduit alors le résultat par calcul immédiat du logarithme discret en base g.

Sécurité : Le problème de déchiffrement se réduit par construction à celui du déchiffrement du système RSA quadratique et à celui de la factorisation. Comme pour le système de Galindo *et al.* on ne sait pas s'il y a équivalence polynomiale entre les deux problèmes.

Le théorème II – 17 s'applique encore ici. La sécurité sémantique du schéma repose donc une nouvelle fois sur la difficulté de la reconnaissance des éléments de $f(\Lambda)$ dans Ω.

Efficacité : L'expansion du cryptosystème est de 4 *a priori*, mais elle peut être réduite à 3, comme vu en sous-section 2.3.

Pour le chiffrement, l'évaluation de la fonction f se fait en $9|e|_2$ multiplications modulo n en utilisant les suites de Lucas (cf. la proposition III – 8). Pour le déchiffrement, on trouve, en utilisant les restes chinois, que les étapes principales utilisent $|n|_2+6|e|_2$ multiplications modulo n.

Généralisation modulo n^{s+1}, $s>1$: Pour généraliser le système avec $s>1$, on posera

$$\Lambda := \left\{(x,y)\in\mathbf{N}^2,\, x<n,\, y<n^{s+1},\, \operatorname{pgcd}(x^2-1,n)=\operatorname{pgcd}(y,n)=1\right\},$$

et

$$\Omega := \left\{(x,y)\in\mathbf{N}^2,\, x<n^{s+1},\, y<n^{s+1},\, \operatorname{pgcd}(x^2-1,n)=\operatorname{pgcd}(y,n)=1\right\}.$$

Pour chiffrer, si e est premier avec $(p^2-1)(q^2-1)$, on utilisera la fonction

$$\mathcal{E}_{\left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^\wedge,\mathrm{RSA\text{-}Q}_{n,e},g} : \begin{cases} \mathbf{Z}/n^s\mathbf{Z} \times \Lambda \longrightarrow \Omega \\ (m\ ,\ (x,y)) \longmapsto (c_1,c_2) \end{cases}$$

où (c_1,c_2) est l'élément de Ω tel que

$$c_1+c_2\sqrt{\Delta} \equiv g^m\rho^e \pmod{n^{s+1}\mathcal{O}_\Delta},$$

avec $\Delta\equiv(x^2-1)y^{-2} \pmod{n^{s+1}}$, $\rho\equiv x+y\sqrt{\Delta} \pmod{n^{s+1}\mathcal{O}_\Delta}$ un élément du groupe $\left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^\wedge$ et g un autre élément de $\left(\mathcal{O}_\Delta/n^{s+1}\mathcal{O}_\Delta\right)^\wedge$, générateur du noyau de la réduction modulaire obtenue comme en sous-section III–3.3. Cette fonction est encore bijective.

Pour le déchiffrement, le relèvement se fait par récurrence en utilisant la proposition III – 9 et le calcul du logarithme discret en base g par l'algorithme donné figure III.5, page 61.

Une seconde adaptation

La première idée d'adaptation de la fonction trappe II–2.1 dans les quotients quadratiques présentée précédemment était assez naturelle : tout comme le système de Catalano *et al.* pouvait être vu comme un RSA probabiliste et le système de Galindo *et al.* un KMOV probabiliste, on a adapté cette fonction trappe générique en utilisant la fonction RSA quadratique.

Cependant, on a vu, en sous-section 1.3 que la fonction LUC offrait la même sécurité que la fonction RSA quadratique, tout en entraînant un coût calculatoire moindre.

Dans cette sous-section, on montre donc comment utiliser la fonction trappe II–2.1 avec la fonction LUC décrite page 73. Ce cryptosystème a été exposé dans [Cas].

Cadre : Le système que l'on va décrire a en fait pour cadre le groupe $G := \left(\mathbf{Z}/n^2\mathbf{Z}\right)^\times$, comme le système de Catalano *et al.*, même si on utilise la fonction LUC dont les propriétés sont issues de groupes du type $\left(\mathcal{O}_\Delta/n^2\mathcal{O}_\Delta\right)^\wedge$. On utilise de nouveau des groupes fixes, et on se situe donc directement dans le cadre de la sous-section II–2.1.

On note $H := (\mathbf{Z}/n\mathbf{Z})^\times$ de telle sorte que l'élément $g \equiv 1+n \pmod{n^2}$ soit un générateur du noyau (d'ordre n) de la surjection canonique de G sur H.

Les ensembles Ω et Λ sont choisis comme suit :

$$\Omega := \left\{x \in \mathbf{N},\, x < n^2,\, \mathrm{pgcd}(x^2-4) = \mathrm{pgcd}(x,n) = 1\right\},$$

et

$$\Lambda := \left\{x \in \mathbf{N},\, x < n,\, \mathrm{pgcd}(x^2-4) = \mathrm{pgcd}(x,n) = 1\right\}.$$

Soit e un entier premier avec $(p^2-1)(q^2-1)$. D'après la proposition IV – 5, la fonction $\mathrm{LUC}_{n,e}$ est une permutation de Λ. Avec les notations du paragraphe « Construction de la fonction f » de la page 26, on a en fait $\overline{\Lambda} = \Lambda$ et on pose $\overline{f} = \mathrm{LUC}_{n,e}$. On relève alors cette fonction de Λ dans Ω en la fonction

$$f : x \longmapsto (\mathrm{V}_e(x) \bmod n^2).$$

Fonction de chiffrement : On considère la fonction de chiffrement suivante.

$$\mathcal{E}_{(\mathbf{Z}/n^2\mathbf{Z})^\times,\mathrm{LUC}(n,e),g} : \begin{cases} \mathbf{Z}/n\mathbf{Z} \times \Lambda & \longrightarrow & \Omega \\ (m\quad ,\quad r) & \longmapsto & g^m \mathrm{V}_e(r) \bmod n^2 \end{cases}$$

où m est le message clair et r un aléa.

Par construction cette fonction est bien définie et c'est une bijection.

Clef privée et déchiffrement : La clef privée du cryptosystème est la trappe permettant d'inverser $\overline{f} = \mathrm{LUC}_{n,e}$. Le déchiffrement se fait toujours comme indiqué en sous-section II–2.1.

On donne dans la figure IV.1 un algorithme de déchiffrement utilisant les restes chinois. Pour avoir un coût réduit, on utilise comme clef de déchiffrement le vecteur

$$\left(d_{(p,-1)}, d_{(p,1)}, d_{(q,-1)}, d_{(q,1)}\right),$$

où $d_{(p,i)} \equiv e^{-1} \pmod{p-i}$ pour $i = \pm 1$, et de même pour le nombre premier q, en notant p et q les facteurs premiers de n.

Vérifions que l'algorithme de la figure IV.1 retourne bien un message valide étant donné un chiffré c. On suppose que $c = g^m \mathrm{V}_e(r) \bmod n^2$. La première partie de l'algorithme permet de retrouver $r \in \Lambda$. Soit Δ un entier non carré tel que $\Delta \equiv r^2 - 4 \pmod{n^2}$ et $\alpha \in \mathcal{O}_\Delta$ tel que $\alpha \equiv \frac{r+\sqrt{\Delta}}{2} \pmod{n^2\mathcal{O}_\Delta}$.

On a $\mathrm{N}(\alpha) \equiv 1 \pmod{n^2}$ et $\mathrm{V}_e(r) \equiv \mathrm{Tr}(\alpha^e) \pmod{n^2}$. D'après la proposition III – 8, on a $4\mathrm{N}(\alpha^e) \equiv \mathrm{V}_e(r)^2 - \Delta \mathrm{U}_e(r)^2 \equiv 4 \pmod{n^2}$, la dernière congruence découlant du fait que $\mathrm{N}(\alpha^e) \equiv 1 \pmod{n^2}$. On a donc

$$\Delta \equiv \frac{\mathrm{V}_e(r)^2 - 4}{\mathrm{U}_e(r)^2} \pmod{n^2}.$$

Déchiffrement

entrée : (p,q,d) la clef privée, e l'exposant public et c un chiffré

pré-calculs : $inv_q := p^{-1} \bmod q$, $inv_p := q^{-1} \bmod p$,
$pre_{\text{CRT}} := p^{-1} \bmod q$.

pour $l \in \{p,q\}$ **faire**
$i := \left(\frac{c^2-4}{l}\right)$,
$r_l := \mathrm{V}_{d_{(l,i)}}(c) \bmod l$.
fin
$r := r_p + p(r_q - r_p)pre_{\text{CRT}} \bmod pq$

pour $l \in \{p,q\}$ **faire**
$tmp := c/\mathrm{V}_e(r) \bmod l^2$,
$tmp := (tmp - 1)/l$,
$m_p :\equiv tmp \times inv_l \bmod l$.
fin

sortie : $m :\equiv m_p + p(m_q - m_p)pre_{\text{CRT}} \pmod{pq}$
/* le clair de $\mathbf{Z}/n\mathbf{Z}$ correspondant à c. */

FIG. IV.1 – Algorithme de déchiffrement utilisant les restes chinois

Dans cette équation, $\mathrm{U}_e(r)$ est inversible car, si $r \in \Lambda$, par définition, Δ est premier avec n et d'après la proposition IV – 4, $\mathrm{pgcd}(\mathrm{V}_e(r)^2 - 4, n) = 1$.

Cette expression de Δ permet de retrouver la valeur du symbole de Legendre :

$$\left(\frac{\Delta}{p}\right) = \left(\frac{\mathrm{V}_e(r)^2 - 4}{p}\right) = \left(\frac{c^2-4}{p}\right).$$

Avec cette valeur, l'algorithme utilise l'inverse d_p de e modulo $\varphi_\Delta(p)$ et retrouve $r_p := \mathrm{V}_{d_p}(c) \bmod p = \mathrm{V}_{d_p}(\mathrm{V}_e(r)) \bmod p = r \bmod p$. L'algorithme calcule de même la valeur $r_q := r \bmod q$. Pour retrouver $(m \bmod p)$ à partir de $(c \bmod p^2)$, on doit connaître la valeur $(r \bmod p^2)$. Étant donnés r_p et r_q on retrouve $(r \bmod n)$ grâce au théorème des restes chinois :

$$r := r_p + p(r_q - r_p)pre_{\text{CRT}} \bmod pq.$$

Comme r a été choisi plus petit que n, on retrouve en fait la valeur de r dans $\mathbf{Z}$, et on en déduit les valeurs r modulo p^2 et q^2.

On a $c/\mathrm{V}_e(r) \equiv (1+(mq)p) \pmod{p^2}$. On peut donc calculer $mq \bmod p$ et $m \bmod p$. L'algorithme effectue les mêmes calculs modulo q et la valeur de m modulo n en est déduite par une autre application du théorème des restes chinois.

Sécurité : Par construction, le problème de déchiffrement se réduit au problème de déchiffrement du système LUC et au problème de la factorisation. Comme pour le système de Galindo *et al.*, on ne sait pas s'il y a équivalence entre les deux problèmes.

D'après le théorème II – 17, la sécurité sémantique du schéma repose sur la difficulté de la reconnaissance des éléments de $f(\Lambda)$ dans Ω.

Efficacité : Comme le système de Catalano, ce système a une expansion de 2. Elle peut être réduite en généralisant de même manière que le système de Catalano ce système dans $(\mathbf{Z}/n^{s+1}\mathbf{Z})^\times$.

Pour le chiffrement, l'évaluation de la fonction f se fait en $6|e|_2$ multiplications modulo n en utilisant l'algorithme de la figure III.3 page 55.

Pour le déchiffrement, on trouve, en utilisant les restes chinois, que les étapes principales utilisent $\frac{2}{3}|n|_2 + 4|e|_2$ multiplications modulo n.

3.4. Comparaison des systèmes

On résume dans le tableau qui suit les caractéristiques des systèmes probabilistes non homomorphiques présentés précédemment dans le cas $s = 1$.

Pour le coût des cryptosystèmes, on compte toutes les opérations et non pas seulement les plus coûteuses. Le schéma proposé en sous-section 3.1 (resp. en sous-section 3.3) est dénommé LUC-RSA (resp. LUC proba).

Cryptosystème	Catalano *et al.*	Galindo *et al.*	LUC proba	LUC-RSA	El Gamal EC
Cadre	$(\mathbf{Z}/n^2\mathbf{Z})^\times$	E_{n^2}	$(\mathbf{Z}/n^2\mathbf{Z})^\times$	$(\mathbf{Z}/n^2\mathbf{Z})^\times$	E_p
Taille de l'entrée	$\lvert n\rvert_2$				$2\lvert p\rvert_2$
Taille de la sortie	$2\lvert n\rvert_2$	$3\lvert n\rvert_2$	$2\lvert n\rvert_2$		$4\lvert p\rvert_2$
Expansion	2	3	2		
Chiffrement	$\frac{9}{2}\lvert e\rvert_2 + 1$	$35\lvert e\rvert_2 + 3$	$6\lvert e\rvert_2 + 1$	$\frac{7}{2}\lvert e\rvert_2 + 11$	$40\lvert p\rvert_2 + 13$
Déchiffrement	$\frac{\lvert n\rvert_2}{2} + 3\lvert e\rvert_2 + 19$	$\frac{13}{2}\lvert n\rvert_2 + 35\lvert e\rvert_2 + 24$	$\frac{2}{3}\lvert n\rvert_2 + 4\lvert e\rvert_2 + 18$	$\frac{7}{6}\lvert n\rvert_2 + 1$	$20\lvert p\rvert_2 + 13$

Dans le tableau, on a utilisé les estimations habituelles : une multiplication modulo n^2 est jugée aussi coûteuse que 3 multiplications modulo n ; trois multiplications modulo p autant qu'une modulo n et une multiplication modulo p^2 autant qu'une multiplication modulo n. L'unité est la multiplication modulo n. Une inversion modulo n a été jugée aussi coûteuse que 10 inversions modulo n. Les systèmes faisant intervenir la fonction LUC ont besoin de plus d'effectuer deux calculs de symbole de Legendre dans $\mathbf{Z}/p\mathbf{Z}$ avec $|p|_2 = |n|_2/2$ pour le déchiffrement. Ce calcul est de complexité $\mathcal{O}(|p|_2^2)$.

Pour pouvoir bien comparer ces systèmes, on va utiliser des tailles de clefs donnant le même niveau de sécurité. Les quatre premiers systèmes utilisent un entier RSA n et un exposant public e. Comme la sécurité de tous ces systèmes est basée sur le problème de la factorisation de n, on choisit un n de même taille pour tous ces systèmes. On prend $|n|_2 = 1024$.

D'après la discussion de la section 1.3, l'exposant public e doit avoir la même taille pour le système de Catalano *et al.* et pour les systèmes basés sur LUC. Pour le système de Galindo *et al.*, on peut utiliser un exposant de taille deux fois moindre. On prend un entier de 16 bits pour le système LUC proba (à cause de l'algorithme de la figure III.3 page 55, un entier spécial avec faible poids de Hamming ne permet pas d'accélérer le calcul), l'exposant spécial $2^{16}+1$ (resp. 2^4+1) est pris pour les systèmes de Catalano *et al.* et LUC-RSA (resp. pour le système de Galindo *et al.*) et la forme de l'exposant est prise en considération pour déterminer le nombre d'opérations nécessaires pour les exponentiations.

Pour le système El Gamal elliptique, étant donnée la difficulté du problème du logarithme discret dans les courbes elliptiques, un nombre premier de 192 bits suffit pour avoir un niveau de sécurité aussi fort qu'avec un module RSA de 1024 bits dans le cadre de systèmes basés sur la factorisation. On résume les coûts correspondants dans le tableau suivant où l'unité est toujours la multiplication modulo n.

Cryptosystème	Catalano	Galindo	El Gamal EC	LUC proba	LUC-RSA
Taille de l'entrée	1024	1024	384	1024	
Taille de la sortie	2048	3072	384	2048	
Chiffrement	52	125	225	97	**49**
Déchiffrement	**565**	6952	113	765	1196
Clef publique	1040	1032	1344	1040	
Clef privée	1040	1032	768	1040	

L'utilisation des suites de Lucas amène un léger surplus au niveau du coût calculatoire. Ainsi, le système LUC proba est légèrement plus lent que le système de Catalano *et al.*. Le système LUC-RSA qui travaille modulo n se révèle être le plus efficace en chiffrement. Par contre son déchiffrement est pénalisé par les deux exponentiations modulaires complètes nécessaires.

Le système de Galindo *et al.*, même s'il demeure plus lent que les autres schémas, concurrence le système El Gamal elliptique en chiffrement (si l'on calcule le nombre d'opérations nécessaires par bits chiffrés on trouve $0,122$ pour Galindo *et al.* et $0,586$ pour El Gamal). Par contre son déchiffrement est catastrophique. Notons que le système LUC proba est aussi largement plus rapide qu'El Gamal EC en chiffrement et que le rapport nombre d'opérations pour chiffrer et déchiffrer par le nombre de bits d'entrée est légèrement en faveur du système LUC proba ($0,841$ contre $0,889$).

Ainsi, si l'on veut utiliser un système probabiliste dont la sécurité n'est pas basée sur RSA le système LUC proba doit être sérieusement pris en compte.

BIBLIOGRAPHIE

[AD82] L. M. ADELMAN et R. Mc DONNELL : An Application of Higher Reciprocity to Computational Number Theory. *In Proc. of 23rd IEEE Symp. on Foundations of Computer Science*, p. 100-106, 1982.

[AM93] A. O. L. ATKIN et F. MORAIN : Elliptic curves and primality proving. *Mathematics of Computation*, vol. 61, n° 203, p. 29-68, 1993.

[Arn88] François ARNAULT : *Sur quelques tests probabilistes de primalité*. Thèse de doctorat, Université de Poitiers, 1988.

[BBL95] D. BLEICHENBACHER, W. BOSMA et A. K. LENSTRA : Some remarks on Lucas-based cryptosystems. *In Proc. of Crypto' 95*, p. 386-396, 1995.

[BDJR97] M. BELLARE, A. DESAI, E. JOKIPII et P. ROGAWAY : A Concrete Security Treatment of Symmetric Encryption. *In Proc. of FOCS' 97*, p. 394-405, 1997.

[BDPR98] M. BELLARE, A. DESAI, D. POINTCHEVAL et P. ROGAWAY : Relations Among Notions of Security for Public-Key Encryption Schemes. *In Proc. of Crypto' 98*, p. 26-45, 1998.

[Ben88] Josh C. BENALOH : *Verifiable Secret-Ballot Elections*. Thèse de doctorat, Yale University, 1988.

[BJ02] E. BRIER et M. JOYE : Weierstraß Elliptic Curves and Side-Channel Attacks. *In Proc. of PKC' 02*, p. 335-345, 2002.

[BSSW80] R. BAILLIE et Jr. S. S. WAGSTAFF : Lucas pseudoprimes. *Mathematics of Computation*, vol. 35, n° 152, p. 1391-1417, 1980.

[Cas] Guilhem CASTAGNOS : An efficient probabilistic public-key cryptosystem over quadratic fields quotients. À paraître *in Finite Fields and Their Applications*.

[CGHGN01] D. CATALANO, R. GENNARO, N. HOWGRAVE-GRAHAM et P. Q. NGUYEN : Paillier's cryptosystem revisited. *In Proc. of CCS' 01*, p. 206-214. ACM Press, 2001.

[CGS97] R. CRAMER, R. GENNARO et B. SCHOENMAKERS : A Secure and Optimally Efficient Multi-Authority Election Scheme. *In Proc. of Eurocrypt' 97*, p. 103-118, 1997.

[CNS02] D. CATALANO, P. Q. NGUYEN et J. STERN : The Hardness of Hensel Lifting : The Case of RSA and Discrete Logarithm. *In Proc. of Asiacrypt' 02*, p. 299-310, 2002.

[Coh93] Henri COHEN : *A course in Computational Algebraic Number Theory*. Springer-Verlag, 1993.

[Cop96] D. COPPERSMITH : Finding a Small Root of a Univariate Modular Equation. *In Proc. of Eurocrypt' 96*, p. 155-165, 1996.

[dB53] N. G. de BRUIJN : On the factorization of cyclic groups. *Indagationes Math.*, vol. 15, p. 370-377, 1953.

[Des86] M. DESBOVES : Résolution, en nombres entiers et sous sa forme la plus générale, de l'équation cubique, homogène, à trois inconnues. *Ann. de Mathématiques*, vol. 45, p. 545-579, 1886.

[DJ01] I. DAMGÅRD et M. J. JURIK : A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System. *In Proc. of PKC' 01*, p. 119-136, 2001.

[dS00] Bart de SMIT : The cyclic subfield integer index. *Journal de Théorie des Nombres de Bordeaux*, vol. 12, p. 209-218, 2000.

[Gal02] Steven D. GALBRAITH : Elliptic Curve Paillier Schemes. *Journal of Cryptology*, vol. 15, n° 2, p. 129-138, 2002.

[GH99] G. GONG et L. HARN : Public-Key Cryptosystems Based on Cubic Finite Field Extensions. *In IEEE Trans. Inform. Theory*, vol. 45, p. 2601-2605, 1999.

[GHW01] G. GONG, L. HARN et H. WU : The GH public-key cryptosystem. *In Proc. of SAC' 01*, p. 284-300, 2001.

[GM84] S. GOLDWASSER et S. MICALI : Probabilistic Encryption. *JCSS*, vol. 28, n° 2, p. 270-299, 1984.

[GMMV02] D. GALINDO, S. MARTÍN, P. MORILLO et J. L. VILLAR : An Efficient Semantically Secure Elliptic Curve Cryptosystem Based on KMOV. *In Proc. of WCC' 03*, p. 213-221, 2002.

[HS00] M. HIRT et K. SAKO : Efficient Receipt-Free Voting Based on Homomorphic Encryption. *In Proc. of Eurocrypt' 00*, p. 539-556, 2000.

[JQ96] M. JOYE et J.-J. QUISQUATER : Efficient computation of full Lucas sequences. *In Electronics Letters*, vol. 32, p. 537-538, 1996.

[JQ98] M. JOYE et J.-J. QUISQUATER : Cryptanalysis of RSA-type cryptosystems : a visit. *In* R. N. WRIGHT et P. G. NEUMANN, éditeurs : *Network Threats*, p. 21-31. American Mathematical Society, 1998.

[JQ01] M. JOYE et J.-J. QUISQUATER : Hessian Elliptic Curves and Side-Channel Attacks. *In Proc. of CHES' 01*, p. 402-410, 2001.

[KK98] N. KUNIHIRO et K. KOYAMA : Equivalence of Counting the Number of Points On Elliptic Curve over the Ring $\mathbf{Z}/n\mathbf{Z}$ and Factoring n. *In Proc. of Eurocrypt' 98*, p. 47-58, 1998.

[KMOV91] K. KOYAMA, U. M. MAURER, T. OKAMOTO et S. A. VANSTONE : New Public-Key Schemes Based on Elliptic Curves over the Ring $\mathbf{Z}/n\mathbf{Z}$. *In Proc. of Crypto' 91*, p. 252-266, 1991.

[Kob84] Neal KOBLITZ : *p-adic Numbers, p-adic Analysis, and Zeta-Functions*. Springer, 1984.

[Len87a] Hendrik W. LENSTRA, JR. : Elliptic Curves and Number Theoretic Algorithms. *In* AMS, éditeur : *Proc. of Internatinal Congr. Math.*, p. 99-120, 1987.

[Len87b] Hendrik W. LENSTRA, JR. : Factoring integers with elliptic curves. *Annals of Mathematics*, vol. 126, n° 2, p.649-673, 1987.

[LM84] R. LIDL et W. B. MULLER : Permutation Polynomials in RSA Cryptosystems. *In Proc. of Crypto' 83*, p. 293-301, 1984.

[LMT93] R. LIDL, G. L. MULLEN et G. TURNWALD : *Dickson Polynomials*, vol. 65 de *Pitman Monographs and Surveys in Pure and Applied Mathematics*. Longman Scientific & Technical, 1993.

[LN97] R. LIDL et H. NIEDERREITER : *Finite fields*, vol. 20 de *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 1997.

[LV00] A. K. LENSTRA et E. R. VERHEUL : The XTR public key system. *In Proc. of Crypto' 00*, p. 1-19, 2000.

[Men93] Alfred J. MENEZES : *Elliptic Curve Public Key Cryptosystems*. Kluwer Academic Publishers, 1993.

[MN81] W. B. MÜLLER et R. NÖBAUER : Some remarks on public-key cryptosystems. *Sci. Math. Hungar.*, vol. 16, p. 71-76, 1981.

[MN86] W. B. MÜLLER et R. NÖBAUER : Cryptanalysis of the Dickson-scheme. *In Proc. of Eurocrypt' 85*, p. 50-61. Springer-Verlag, 1986.

[Mor95] Willi MORE : Fast Evaluation of Rédei Functions. *Appl. Algebra Eng. Commun. Comput.*, vol. 6, p. 171-173, 1995.

[MRS88] S. MICALI, C. RACKOFF et B. SLOAN : The notion of security for probabilistic cryptosystems. *SIAM J. Comput.*, vol. 17, n° 2, p. 412-426, 1988.

[NS98] D. NACCACHE et J. STERN : A New Public Key Cryptosystem Based on Higher Residues. *In Proc. of CCS' 98*, p. 59-66, 1998.

[OU98a] T. OKAMOTO et S. UCHIYAMA : A New Public Key Cryptosystem as Secure as Factoring. *In Proc. of Eurocrypt' 98*, p. 308-318, 1998.

[OU98b] T. OKAMOTO et S. UCHIYAMA : Security of an Identity-Based Cryptosystem and the Related Reductions. *In Proc. of Eurocrypt' 98*, p. 546-560, 1998.

[Pai99] Pascal PAILLIER : Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. *In Proc. of Eurocrypt' 99*, p. 223-238, 1999.

[Poi99] David POINTCHEVAL : New Public Key Cryptosystems Based on the Dependent-RSA Problems. *In Proc. of Eurocrypt' 99*, p. 239-254, 1999.

[PP99] P. PAILLIER et D. POINTCHEVAL : Efficient Public-Key Cryptosystems Provably Secure Against Active Adversaries. *In Proc. of Asiacrypt' 99*, p. 165-179, 1999.

[PS97] V. PRASOLOV et Y. SOLOVYEV : *Elliptic Functions and Elliptic Integrals*, vol. 170. Translations of Mathematical Monographs, 1997.

[RS03] K. RUBIN et A. SILVERBERG : Torus-Based Cryptography. *In Proc. of Crypto' 03*, p. 349-365, 2003.

[Sil86] Joseph H. SILVERMAN : *The arithmetic of elliptic curves*. Springer, 1986.

[SL93] P. SMITH et M. J. J. LENNON : LUC : A new public key system. *In Proc. of the Ninth IFIP Int. Symp. on Computer Security*, p. 103-117, 1993.

[SS94] P. SMITH et C. SKINNER : A Public-Key Cryptosystem and a Digital Signature System Based on the Lucas Function Analogue to Discrete Logarithms. *In Proc. of ASIACRYPT' 94*, p. 357-364, 1994.

[Var88] Vijay VARADHARAJAN : Permutation Polynomials based Cryptosystems. *International Journal of Computer Mathematics*, vol. 23, p. 237-250, 1988.

[vDGP+05] M. van DIJK, R. GRANGER, D. PAGE, K. RUBIN, A. SILVERBERG, M. STAM et D. WOODRUFF : Practical Cryptography in High Dimensional Tori. *In Proc. of Eurocrypt' 05*, p. 234-250, 2005.

[vDW04] M. van DIJK et D. WOODRUFF : Asymptotically Optimal Communication for Torus-Based Cryptography. *In Proc. of Crypto' 04*, p. 157-158, 2004.

[Wil82] Hugh C. WILLIAMS : A $p+1$ method of factoring. *Mathematics of Computation*, vol. 39, p. 225-234, 1982.

[YL95] S. M. YEN et C. S. LAIH : Fast Algorithms for the LUC Digital Signature Computation. *In IEEE Proceeding of Comput. Digit. Tech*, vol. 142, p. 165-169, 1995.

QUELQUES SCHÉMAS DE CRYPTOGRAPHIE ASYMÉTRIQUE PROBABILISTE

Résumé : Dans cette thèse, on construit de manière générique plusieurs familles de fonctions trappe probabilistes : une famille de fonctions trappe homomorphiques généralisant, entre autres, le cryptosystème de Paillier, et deux autres familles de fonctions trappe, à partir de fonctions trappe déterministes. Pour utiliser ces fonctions trappe, on étudie plusieurs groupes finis : les quotients de $\mathbf{Z}$, les courbes elliptiques définies sur $\mathbf{Z}/n\mathbf{Z}$, où n est un entier impair, pour lesquelles on donne un système complet de formules d'additions, et un autre groupe fini peu utilisé en cryptographie, celui des éléments de norme 1 d'un corps quadratique modulo n. On expose plusieurs cryptosystèmes avec une analyse de leur sécurité et de leur complexité, en utilisant les familles de fonctions trappe dans ces groupes. Dans les quotients de $\mathbf{Z}$ et dans les courbes elliptiques, on retrouve de nombreux cryptosystèmes décrits ces dernières années. Dans les quotients de corps quadratiques, on propose plusieurs nouveaux systèmes probabilistes très performants.

Mots-clefs : Cryptographie probabiliste, chiffrement homomorphique, cryptosystème de Paillier, courbes elliptiques sur des anneaux, quotients de corps quadratiques, cryptosystème LUC, suites de Lucas.

SOME ASYMMETRIC CRYPTOGRAPHY PROBABILISTIC SCHEMES

Abstract: In this thesis, we build, in a generic way, several families of probabilistic trapdoor functions. First, a family of homomorphic trapdoor functions which generalize, among others, the Paillier cryptosystem, and then two others families of trapdoor functions, built from deterministic trapdoor functions. We consider then several finite groups in order to use these trapdoor functions: quotients of $\mathbf{Z}$, elliptic curves over $\mathbf{Z}/n\mathbf{Z}$ (with n odd integer), for which we give a complete set of addition formulæ, and another finite group, not widely used in cryptography, the group of norm 1 elements of a quadratic field modulo n. We describe several cryptosystems, using the corresponding trapdoor functions in these groups, together with an analysis of their security and their complexity. With quotients of $\mathbf{Z}$ and elliptic curves, we get some cryptosystems yet described in the past few years. Using quadratic fields quotients, we propose several new efficient probabilistic schemes.

Keywords: Probabilistic cryptography, homomorphic encryption, Paillier cryptosystem, elliptic curves over rings, quadratic fields quotients, LUC cryptosystem, Lucas sequences.