Sécurité et efficacité des schémas cryptographiques.

Duong Hieu Phan

École normale supérieure
Département d'informatique

École polytechnique
Direction du 3e cycle

Sécurité et efficacité des schémas cryptographiques

THÈSE

présentée et soutenue publiquement le 16 septembre 2005

pour l'obtention du

Doctorat de l'École polytechnique

par

Dương Hiệu PHAN

Composition du jury :

Marc Girault (*Rapporteur*)
Antoine Joux (*Rapporteur*)
Jacques Patarin
David Pointcheval (*Directeur de thèse*)
Guillaume Poupard
Jacques Stern
Jean-Marc Steyaert
Moti Yung

Travaux effectués au Groupe de Recherche en Complexité et Cryptographie (GRECC)
Laboratoire d'informatique de l'ÉNS – 45, rue d'Ulm – 75230 Paris Cedex 05

à mes parents …

« *Và, con ơi,*
Bố muốn tặng con một chú ngựa rơm,
Trên gập ghềnh đường xa
con sẽ bước… »
(Phan Đình Diệu*, Đan Mạch, 1982*)

Remerciements

Ce travail n'aurait jamais vu le jour sans l'aide précieuse de David Pointcheval. David m'a fait découvrir les « preuves de sécurité » et m'a guidé tout au long de ces années de thèse par ses explications aussi claires que précises. Qu'il trouve ici la profonde gratitude de son premier thésard.

Ma reconnaissance s'adresse également à Jacques Stern qui m'a initié à la cryptographie au travers des cours du DEA Algorithmique et m'a fait confiance en m'accueillant dans son équipe.

Je tiens à remercier Marc Girault et Antoine Joux d'avoir accepté la lourde tâche de rapporteur. Je remercie aussi Jacques Patarin, Jean-Marc Steyaert, Guillaume Poupard et Moti Yung de me faire l'honneur de participer au jury.

Je voudrais remercier Phong Q.Nguyễn pour ses conseils ; Michel Abdalla et Dario Catalano pour les discussions enrichissantes au bureau et sur le terrain de foot (!).

L'ambiance très sympathique au sein du groupe GRECC m'a toujours aidé dans mes recherches. Je remercie toutes les personnes avec qui j'ai eu de nombreuses discussions intéressantes : Pierre-Alain Fouque, Louis Granboulan, Damien Stehlé, Emmanuelle Bresson, Sébastien Zimmer, Vivien Dubois, Gilles Piret, Sébastien Kunz-Jacques, Nicolas Gama et Eric Levieil. Un grand merci à l'ensemble du personnel technique et administratif du département, tout particulièrement à Joëlle et à Valérie.

De nombreux échanges ont initié plusieurs de mes travaux, je souhaite ainsi remercier Hervé Chabanne, Nelly Fazio, Fabien Laguillaumie, Khánh Q.Nguyễn, Ludovic Perret, Benny Pinkas, Antonio R.Nicolosi, Thomas Sirvent, Đông Vũ Tô et surtout Benoît Chevallier-Mames pour sa contribution.

Je ne saurais oublier le soutien de ma famille. Chaque fin de semaine avec mes sœurs Quỳnh Dương, Hà Dương et leurs familles, c'est comme si on était toujours au Vietnam.

Mes plus jolis remerciements iront à Thùy Linh qui m'a apporté la « preuve de vérité » de la transformation des joies dans la vie en sources d'inspiration dans le travail…

Avant-Propos

Cette thèse présente nos travaux de doctorat [32, 33, 96, 97, 98, 99] portant sur la sécurité prouvée. Elle est organisée en trois parties : les notions de sécurité ; la construction des schémas de chiffrement prouvés sûrs ; et l'application du chiffrement à la diffusion des données chiffrées.

La cryptologie peut être considérée à la fois comme un art ancien et une science nouvelle. Un art ancien car depuis le temps de César, on utilisait déjà des méthodes artisanales comme le décalage alphabétique, les permutations ou les substitutions pour cacher des informations. Une science nouvelle car ce n'est que depuis les années 1970, à la naissance du chiffrement à clef publique, qu'elle est devenue un thème de recherche académique lié à beaucoup d'autres domaines scientifiques, en particulier à la théorie des nombres, à l'algèbre, à la complexité, à la théorie de l'information, aux codes correcteurs d'erreurs...

La cryptologie, étymologiquement la science du secret, englobe la cryptographie, l'art des écritures cachées, et la cryptanalyse dont le but n'est autre que d'attaquer les méthodes cryptographiques. Les deux domaines ont été séparés jusqu'au milieu du siècle dernier : les cryptographes construisaient d'abord les schémas et les cryptanalystes essayaient de les casser. À la fin des années 40, Shannon, le père de la théorie de l'information, a fait le premier pas vers la cryptologie moderne [113]. Il a proposé la première notion de sécurité du point de vue de la théorie de l'information : un schéma de chiffrement est parfaitement sûr si les textes chiffrés ne contiennent aucune information sur les textes clairs. Cette notion a aussi, pour la première fois, fait un pont entre la cryptographie et la cryptanalyse : un schéma est sûr s'il n'existe pas d'attaque, même à puissance illimitée, capable de révéler la moindre information sur le texte clair à partir du texte chiffré. Cette notion de sécurité parfaite est très forte, elle exige en effet que le chiffrement soit symétrique : le secret est partagé entre au moins deux personnes et utilisé à la fois pour le chiffrement et pour le déchiffrement.

L'évolution de la cryptologie, ou la naissance de la cryptographie moderne, est notamment marquée par l'article de Diffie et Hellman [41] en 1976. Leur idée lumineuse consiste à dire que la clef de chiffrement ne doit pas être identique à la clef de déchiffrement (d'où le terme de « cryptographie asymétrique »). On peut donc publier la clef de chiffrement et par conséquent, supprimer le problème du transfert des clefs secrètes. Évidemment, la notion de sécurité parfaite de Shannon n'est plus applicable à la cryptographie asymé-

trique : n'importe qui, étant donné un texte chiffré, à l'aide du chiffrement à clef publique, peut retrouver le texte clair correspondant en faisant une recherche exhaustive sur tous les textes clairs. Mais pourquoi utiliser une notion aussi forte que la sécurité parfaite ? Pourquoi donner à l'attaquant une puissance illimitée qui n'existe pas en réalité ? On peut se contenter des schémas où toute attaque exige des ressources inatteignables. De cette idée vient la notion de sécurité calculatoire : un schéma est sûr s'il n'existe pas d'attaquant « réalisable » qui peut retrouver le texte clair à partir du texte chiffré. Dans ce cas, l'attaquant, dont la puissance est limitée, est souvent formalisé par une machine de Turing polynomiale et probabiliste.

L'article de Diffie et Hellman a ouvert la voie au développement de la cryptographie moderne. D'un côté, la notion de chiffrement à clef publique conduit à l'apparition de plusieurs nouvelles branches de la cryptographie non moins importantes : la signature, l'identification... De l'autre côté, la notion de sécurité calculatoire devient l'approche la plus utilisée pour étudier la sécurité ; elle a relié non seulement la cryptographie à la cryptanalyse mais aussi la cryptologie à la théorie de la complexité.

Aujourd'hui, les cryptographes peuvent être en quelque sorte aussi considérés comme des cryptanalystes : ils proposent de nouveaux schémas, et essaient aussi de les prouver sûrs en montrant l'inexistence d'attaques. La méthode souvent utilisée est le raisonnement par « l'absurdité ». S'il existe une attaque contre le schéma proposé, on sera alors capable de résoudre un problème difficile au sens de la théorie de la complexité. Une telle preuve est appelée une preuve de sécurité. Une nouvelle branche de la cryptographie est née : la sécurité prouvée.

La sécurité prouvée devient de plus en plus importante en cryptographie. Plusieurs schémas ne possédant pas de preuves de sécurité ont été attaqués ; ce qui confirme la pertinence de cette branche de recherche. Dans la cryptographie actuelle, une nouvelle construction va souvent de paire avec une preuve de sécurité. Les chercheurs de cette jeune branche de recherche ont commencé d'abord par se poser des questions naturelles : Qu'est-ce que la sécurité ? Comment modéliser une attaque avec une puissance donnée ? Ainsi, pour obtenir une preuve de sécurité, il faut d'abord formaliser mathématiquement plusieurs notions telles que la sécurité, le modèle d'attaque avec différentes puissances internes et externes, les scénarios d'attaque, etc. La sécurité prouvée est en étroite relation avec les trois principaux mouvements de la cryptographie : la formalisation des notions de sécurité, la construction des schémas formellement prouvés sûrs et la recherche de nouvelles fonctionnalités pour la cryptographie. Nos travaux ont pour objectif d'étudier la sécurité prouvée, nos principaux résultats seront présentés dans cette thèse.

Dans la première partie, nous rappelons l'histoire des formalisations des notions de sécurité et des modèles d'attaques et présentons brièvement les constructions bien connues ayant atteint des standards de sécurité. Nous présentons ensuite nos résultats (publiés dans deux articles [99, 97]) sur les études des notions de sécurité aussi bien pour le chiffrement asymétrique que pour le chiffrement symétrique. Premièrement, nous étudions plus en détail les modèles d'attaque pour le chiffrement asymétrique, mettons en plus en évidence la relation entre deux notions principales de la sécurité : la sécurité sémantique et la non-

malléabilité. Deuxièmement, nous proposons et considérons une version de la sécurité sémantique pour le chiffrement par bloc et prouvons la relation entre cette notion et les notions conventionnelles comme les permutations (super) pseudo-aléatoires.

Dans la deuxième partie, nous présentons nos résultats (publiés dans trois articles [96, 98, 33]) qui consistent à proposer de nouvelles constructions pour le chiffrement et la signature dans le modèle de la permutation aléatoire et dans le modèle de l'oracle aléatoire. Dans ces modèles, les chiffrements par bloc ou les fonctions de hachage sont supposés parfaitement aléatoires. Nous introduisons ainsi les premiers chiffrements prouvés sûrs et et sans redondance : tout texte chiffré est valide et correspond au chiffrement d'un texte clair. Remarquons que la redondance était auparavant essentielle pour prouver la sécurité des schémas de chiffrement. Nos schémas sont efficaces et peuvent être utilisés non seulement avec les permutations à sens-unique à trappe mais aussi avec une grande classe de fonctions à sens-unique à trappe, y compris les plus connues, telles qu'El-Gamal et Paillier. À la fin de cette partie, nous appliquons aussi notre construction au paradigme des « paddings universels » et obtenons les schémas les plus efficaces de cette catégorie. Un padding universel, introduit par Coron *et. al* [36], permet d'utiliser d'une manière très pratique le même padding et le même couple de clef secrète/clef publique pour gérer à la fois le chiffrement et la signature.

La troisième partie concerne l'application du chiffrement au problème de la diffusion de données chiffrées, la télévision payante par exemple. Dans ce modèle, chaque usager dispose d'un décodeur légitime pour déchiffrer le texte chiffré émis par un centre. Un des plus grands soucis du centre est d'empêcher les usagers de fabriquer de nouveaux décodeurs pirates. Nous étudions le problème du traçage de traîtres qui permet au centre, à partir d'un décodeur pirate, de retrouver ceux qui ont triché (les traîtres). Ce chapitre présente notre article [32]. Nous introduisons une nouvelle fonctionnalité : la traçabilité publique. Cette fonctionnalité permet non seulement au centre mais aussi à n'importe qui, disposant seulement de la clef publique, de faire le traçage de traîtres. Nous proposons d'abord un schéma de base (pour deux usagers) avec cette propriété et nous le généralisons ensuite au cas de plusieurs usagers. Le cas général atteint la traçabilité dans la phase interactive avec le décodeur pirate, qui est la plus coûteuse. Notre schéma améliore aussi l'efficacité du schéma de Kiayias et Yung [73], considéré jusque là comme le meilleur schéma du point de vue de l'optimisation du rapport entre la taille de texte chiffré et celle du texte clair. Dans nos constructions, nous utilisons un outil largement utilisé actuellement : les couplages. Parmi tous les schémas de traçage de traîtres fondés sur les couplages, il ne restait que le schéma de To *et. al* [117] qui n'avait pas été cassé. Nous présentons une attaque sur ce schéma ; notre schéma devient donc le seul schéma qui ne soit pas cassé.

Table des matières

Sécurité prouvée

Chapitre 1
Introduction à la sécurité prouvée

Chapitre 8
Traçabilité publique dans le traçage de traîtres

Table des figures

Notations

$\mathbb{N}$	Ensemble des entiers naturels
$\mathbb{Z}$	Ensemble des entiers relatifs
$\mathbb{Z}_n$	Anneau des classes de résidus modulo n
$\mathbb{Z}_n^\star$	Groupe des éléments inversibles de $\mathbb{Z}_n$ selon la loi de multiplication
$\mathbb{R}, \mathbb{R}^+$	Ensemble des nombres réels, réels positifs (resp.)
1^n	Entier n écrit sous forme unaire
$x \leftarrow \mathcal{A}(...)$	L'algorithme $\mathcal{A}$ génère x
$x \xleftarrow{R} E$	x choisi dans l'ensemble fini E suivant la distribution uniforme
$x \in_{\mathcal{D}} E$	x tiré dans E suivant la distribution $\mathcal{D}$
$(f_i)_{i \in I}$	La famille de fonctions f_i indexée par i dans l'espace I
(f_i)	La famille de fonctions f_i indexée par i dans un espace adéquat
$\{0,1\}^\ell$	Ensemble des chaines binaires de longueur ℓ
$\{0,1\}^\star$	Ensemble des chaines binaires de longueur quelconque
$\mathcal{A}^{\mathcal{O}}$	Machine de Turing ayant accès à l'oracle $\mathcal{O}$
$\perp$	Symbole signifiant une requête non valide

Sécurité prouvée

1

Introduction à la sécurité prouvée

Un des plus grands soucis des cryptographes est le classement des schémas cryptographiques selon leur niveau de sécurité face aux attaquants. Ce problème a été étudié en profondeur, en particulier à partir de la naissance de la cryptographie à clef publique.

La cryptographie à clef publique introduite par Diffie-Hellman [41] permet un chiffrement à partir de la clef publique du destinataire. Seul ce dernier, muni d'une clef secrète associée à sa clef publique, peut déchiffrer le message. La cryptographie à clef publique est parfois appelée « cryptographie asymétrique » (le terme « asymétrique » vient de la différence entre la clef secrète et la clef publique) et comprend plusieurs domaines importants tels que le chiffrement à clef publique, la signature, l'échange de clef... Une des questions qui se posent suite à l'apparition de ce courant de recherche est de savoir si la cryptographie asymétrique rend obsolète la cryptographie conventionnelle (la cryptographie symétrique). La réponse est non ! Au contraire, elle stimule le développement de la cryptographie symétrique. En effet, n'exigeant pas de conversation au préalable pour convenir d'une clef commune, le chiffrement asymétrique dispose d'un avantage crucial. Cependant, il est très lent par rapport au chiffrement symétrique et ne sert donc pas en pratique au chiffrement de données. En réalité, la transmission de données est généralement assurée par une technique hybride qui utilise à la fois un chiffrement asymétrique et un chiffrement symétrique : par la technique « asymétrique », les parties s'échangent une clef « courte » qui servira ensuite au chiffrement des données de manière symétrique. Dans cette première partie sur « la sécurité prouvée », nous considérons des notions de sécurité aussi bien pour le chiffrement asymétrique que pour le chiffrement symétrique. Avant de présenter nos travaux, nous reviendrons sur la genèse des notions de sécurité et quelques résultats importants les concernant.

1.1 Notions de sécurité pour le chiffrement asymétrique

1.1.1 Sécurité parfaite et sécurité au sens de la complexité

La première étape dans l'évaluation de la sécurité est, bien évidemment, la compréhension de la notion de « sécurité ». À ce jour, on en connaît deux définitions majeures.

La première, au sens de la théorie de l'information, a été initialement évoquée par Shannon [113]. Elle concerne l'« information » sur le texte clair « contenue » dans le texte chiffré : un schéma de chiffrement est *parfaitement sûr* si le texte chiffré ne contient aucune information au sujet du texte clair correspondant. Dans le cadre d'un schéma de chiffrement symétrique, il a été démontré que la sécurité parfaite n'est atteinte que si la clef utilisée est aussi longue que le message. Cette condition est une sérieuse limite en pratique.

La deuxième approche, au sens de la théorie de la complexité, est actuellement utilisée dans l'analyse des schémas de chiffrement. Elle ne s'intéresse pas au contenu du texte chiffré mais met l'accent sur la « difficulté » d'extraire de l'information sur le texte clair à partir du texte chiffré. Cette difficulté est considérée au sens de la complexité et elle est souvent comparée à la difficulté d'un problème bien défini : la factorisation, par exemple.

Remarquons que la notion de « sécurité parfaite » n'est pas applicable au chiffrement asymétrique car le texte chiffré contient toujours de l'information sur le texte clair. En effet, tout attaquant de puissance illimitée peut retrouver le texte clair à partir du texte chiffré grâce à une recherche exhaustive dans l'espace des textes clairs et éventuellement, dans un espace d'aléas (dans le cas d'un chiffrement probabiliste). Dans ce qui suit, nous présentons les différentes notions de sécurité d'un schéma de chiffrement qui sont caractérisées par deux éléments : la difficulté d'extraire de l'information et la puissance dont un attaquant dispose pour casser le schéma.

1.1.2 Réduction en terme de complexité

Dans les premiers schémas de chiffrement à clef publique comme RSA [107] ou Merkle-Hellman [81], la nature de la sécurité n'a pas été prouvée, faute d'une définition de la sécurité. En 1979, Rabin [103] a introduit un schéma où la capacité d'un attaquant d'extraire le message complet à partir d'un texte chiffré est calculatoirement équivalente à la factorisation. Le schéma de chiffrement de Rabin est décrit comme suit : chaque utilisateur choisit deux nombres premiers de même taille p, q (servant de clef secrète) et publie la clef publique $n = pq$. Le chiffrement d'un message m est $c = m^2 \bmod n$. Le receveur, grâce à la connaissance de p, q, peut facilement calculer les quatre racines carrées du chiffrement c et en choisir le message approprié (une façon pratique de conduire à un bon choix est de mettre de la redondance dans le message initial). La sécurité considérée pour ce schéma est dite *à sens-unique* (dénotée OW, par *one-way* en anglais). Un schéma est « à sens-unique » si, à partir d'un *challenge* chiffré, l'attaquant ne peut retrouver le texte

clair correspondant.

Pour prouver la sécurité, on effectue une *réduction* d'un *problème algorithmique* à un *problème de sécurité*. Dans ce schéma, les deux problèmes sont respectivement une solution de la factorisation et l'extraction du message complet à partir d'un texte chiffré. En entrée, une instance n du problème de factorisation est donnée (n est la multiplication de deux premiers). Le but est de résoudre le problème de factorisation pour l'instance n en utilisant un attaquant contre le schéma. L'attaquant, étant capable d'extraire le message complet à partir d'un texte chiffré, joue le rôle de *boîte noire*, *i.e.* il reçoit une entrée et en retourne une sortie sans que l'on ne connaisse le mécanisme. Pour utiliser cet attaquant, il nous faut donc construire un *simulateur* de l'environnement d'attaque. Le simulateur créera un environnement parfaitement similaire (ou au moins indistinguable aux yeux de l'attaquant) à l'environnement réel de l'attaque.

Dans le cas du chiffrement de Rabin, un tel simulateur peut être construit de façon simple. En effet, il publie le nombre n comme la clef publique, puis, choisit un texte aléatoire m et le chiffre comme $c = m^2 \bmod n$. Le simulateur donne la clef publique et le chiffré c à l'attaquant. Comme il s'agit d'un attaquant passif (qui ne demande pas d'information supplémentaire au système), le simulateur n'a pas à simuler les informations à échanger. La simulation est alors parfaite. L'attaquant retournera les quatre racines carrées correspondant au chiffré c. Le simulateur choisit m' parmi ces quatre racines carrées tel que $m' \neq \pm m \bmod n$. Alors, un des deux facteurs de n est le plus petit diviseur commun de n et $m - m'$, d'où la factorisation de n.

Le schéma de Rabin est ainsi le premier système dans lequel une *hypothèse algorithmique* — la difficulté de la factorisation au sens de la complexité — a été réduite à un problème de sécurité — la capacité d'extraire le message complet à partir d'un texte chiffré. Une telle réduction s'appelle aujourd'hui « une preuve de sécurité ».

1.1.3 Fonctions à sens-unique

Dans ce premier exemple, on peut traduire l'« hypothèse algorithmique » de la difficulté de la factorisation en propriété « à sens-unique » de la fonction de multiplication de deux nombres premiers, *i.e.* elle est facile à calculer mais difficile à inverser. Cette notion de *fonction à sens-unique* est fondamentale en cryptographie. Dans les schémas asymétriques, elle est souvent couplée avec une propriété supplémentaire : l'inverse peut être facilement calculé grâce à quelques informations additionnelles, *i.e.* une trappe. Un des candidats pour les permutations à sens-unique à trappe est la fonction RSA où l'inversion est conjecturée difficile mais devient facile avec la connaissance de la factorisation du module utilisé. Une permutation à sens-unique à trappe implique naturellement un schéma OW contre les attaquants passifs. Cependant, la recherche en cryptographie considère d'autres notions de sécurité plus subtiles que la propriété à sens-unique et d'autres types d'attaquants plus puissants qu'un attaquant passif. À ce stade, une question importante est la suivante : « la cryptographie peut-elle être fondée uniquement sur les hypothèses générales telles que l'existence de fonctions (permutations) à sens-unique ou de fonctions

(permutations) à sens-unique à trappe ? ».

1.1.4 Sécurité sémantique et indistinguabilité

Dans les années 80, les chercheurs ont formellement défini les notions de sécurité pour les primitives cryptographiques (notamment pour la signature [60, 61] et pour le chiffrement [59]). Goldwasser et Micali [59] ont introduit la notion de la *sécurité sémantique* et ont proposé la construction d'un chiffrement probabiliste qui peut garantir la sécurité sémantique.

La notion de sécurité sémantique est une notion très forte, elle couvre l'exigence de ne pas pouvoir extraire une information, même partielle (ne serait-ce qu'un seul bit) sur le clair à partir du chiffré. Elle coïncide avec la notion de la sécurité parfaite limitée aux attaques polynomiales probabilistes. La sécurité sémantique est la propriété désirée pour tout schéma de chiffrement, mais, en réalité, on étudie souvent la notion de *l'indistinguabilité* (dénotée IND), plus simple à analyser. Cette notion a également été introduite dans [59]. Goldwasser et Micali [59] (l'indistinguabilité implique la sécurité sémantique) et Micali, Rackoff et Sloan [82] (la sécurité sémantique implique l'indistinguabilité) ont démontré qu'elle était équivalente à la notion de la sécurité sémantique. Avec l'indistinguabilité, l'attaquant ne peut pas trouver deux textes clairs m_0 et m_1 dont il pourrait distinguer les chiffrements : il reçoit un challenge c, chiffré de m_0 ou m_1 et ne doit pas pouvoir deviner à quel texte clair c correspond.

Cette condition implique immédiatement que le chiffrement soit probabiliste. De ce fait, les chiffrements déterministes de RSA et de Rabin n'atteignent pas la sécurité sémantique. Les premiers schémas qui ont atteint cette notion forte sont le schéma de Goldwasser et Micali [59] qui repose sur la difficulté du problème de la résiduosité quadratique, et le schéma de Yao [120], fondé sur l'hypothèse générale de l'existence de fonctions injectives à sens-unique à trappe. Cependant, ces deux schémas sont totalement inefficaces car leurs chiffrements sont « bit par bit », *i.e.* chaque bit est chiffré de manière indépendante, elles conduisent à des chiffrés très larges. Un schéma plus efficace (à chiffrés plus courts) ayant atteint la propriété d'indistinguabilité est celui de Blum et Goldwasser [16]. Il est fondé sur la difficulté du problème de la factorisation. L'idée dans ce schéma est d'utiliser le générateur Blum-Blum-Shub [14] pour générer des bits aléatoires qui masqueront ensuite le clair.

1.1.5 Attaques à chiffrés choisis

Ainsi, l'indistinguabilité a été initialement définie dans le scénario de base où l'attaquant n'a accès qu'à l'information publique et, de ce fait, peut chiffrer des clairs de son choix, d'où le nom d'« attaques à clairs choisis » (dénotée CPA, par *chosen plaintext attack* en anglais). Désormais, on dénote un schéma qui est sûr au sens XXX (IND, par exemple) face aux attaques YYY (CPA, par exemple), un schéma $\mathsf{XXX}-\mathsf{YYY}$ sûr ($\mathsf{IND}-\mathsf{CPA}$ sûr,

par exemple).

Naor et Yung [87] ont étudié la notion d'attaque à chiffrés choisis dans le cas où l'attaquant peut accéder à un oracle (dit l'oracle de déchiffrement) qui retourne, pour chaque chiffré, le clair correspondant.

Face à la puissance de ce type d'attaque, tous les schémas précédents deviennent vulnérables. Naor et Yung [87] ont construit le premier schéma résistant à ce type d'attaque en exploitant la notion de preuve à divulgation de connaissance nulle d'appartenance à un langage (dénotée preuve NIZK, par *non-interactive zero-knowledge proof* en anglais), introduite par Blum, Feldman et Micali [15] (*i.e.* la version non-interactive de la notion de preuve zero-knowledge de l'appartenance à un langage introduite par [58]). Dans [15], Blum *et. al* ont déjà considéré des attaques à chiffrés choisis. Ils ont même suggéré une manière de construire un schéma résistant aux attaques à chiffrés choisis.

Blum *et. al* [15] ont proposé le schéma (sans construction formelle) suivant : au lieu d'envoyer le chiffré $c = E(m)$, il faut envoyer c et σ, une preuve non-interactive zero-knowledge, justifiant que le déchiffrement de c est connu de l'expéditeur. Bien que la notion de NIZK soit proposée dans le même article, remarquons qu'il s'agit plutôt d'une preuve de *connaissance* dans la construction. Le déchiffrement vérifie d'abord si σ est convaincante, auquel cas il retourne m, sinon, il ne retourne rien. De cette façon, l'oracle de déchiffrement semble inutile : pour que l'oracle retourne le message, l'attaquant doit soumettre un chiffré avec une preuve convaincante de connaissance du déchiffrement, *i.e.* il doit déjà connaître le clair. Ainsi, une fois que le chiffrement E est sémantiquement sûr face aux attaques passives, le chiffrement modifié l'est également face aux attaques à chiffrés choisis. Cependant, Blum *et. al* n'ont pas expliqué les étapes de construction d'une telle preuve de connaissance et n'ont pas formellement prouvé la sécurité.

Le schéma proposé par Naor et Yung s'est inspiré de l'idée de Blum *et. al* avec une utilisation judicieuse des preuves NIZK. Le point de départ était toujours un chiffrement E $\mathsf{IND-CPA}$ sûr. Le chiffrement d'un message m retourne un couple $(c_1 = E(e_1, m), c_2 = E(e_2, m))$ avec deux clefs publiques différentes e_1, e_2 (les deux clefs secrètes correspondantes sont d_1, d_2) ainsi qu'une preuve NIZK σ que c_1 et c_2 sont deux chiffrés du même message. Le déchiffrement ne retourne le message que si la preuve est convaincante. L'idée est que la connaissance d'une des deux clefs secrètes d_1, d_2 est suffisante pour le déchiffrement. Alors, le simulateur, en choisissant une de ces deux clefs, disons d_2, pourra bien simuler le déchiffrement et réduire une attaque à chiffrés choisis contre le nouveau schéma à une attaque passive contre $E(e_1)$. Le schéma résiste ainsi aux attaques à chiffrés choisis. Cependant, le modèle d'attaque considéré par Naor et Yung est *à chiffrés choisis non-adaptative* (dénoté $\mathsf{CCA1}$) au sens où les requêtes ne doivent pas dépendre du challenge. En effet, si l'attaquant peut faire des requêtes qui dépendent du challenge, la construction de Naor et Yung ne résiste plus aux attaques à chiffrés choisis. À partir d'une preuve NIZK, il est éventuellement possible de construire une autre preuve NIZK qui ne tient pas compte du dernier bit. Dans ce cas, si (c_1, c_2, σ) est le challenge, alors l'attaquant peut soumettre une requête sur (c_1, c_2, σ'), où σ' ne diffère de σ que par le dernier bit, pour révéler le clair correspondant au challenge. L'attaquant casse ainsi le schéma par

une seule requête adaptative (par rapport au challenge) de déchiffrement. Le schéma de Naor et Yung joue cependant un rôle très important en théorie car c'est le premier schéma à considérer des attaques à chiffrés choisis. De plus, il ne repose que sur une hypothèse générale. En effet, puisque l'existence d'une permutation à sens-unique à trappe est suffisante pour construire un schéma $\mathsf{IND}-\mathsf{CPA}$ sûr [59] et une preuve NIZK [48], elle est aussi suffisante pour le schéma $\mathsf{IND}-\mathsf{CCA1}$ sûr de Naor et Yung.

Le modèle d'attaque *à chiffrés choisis adaptative* (dénoté $\mathsf{CCA2}$) a été étudié pour la première fois par Rackoff et Simon [104]. Dans ce modèle, l'attaquant peut soumettre des requêtes à l'oracle de déchiffrement à tout instant, avant et après la réception du challenge, avec pour seule restriction de ne pas soumettre de requête sur le challenge. Rackoff et Simon ont formalisé la notion de preuve non-interactive zero-knowledge de connaissance, *i.e.* la version non-interactive de la notion de preuve zero-knowledge de connaissance introduite par [46, 47]. En examinant cette dernière, Rackoff et Simon ont proposé un schéma $\mathsf{IND}-\mathsf{CCA2}$ sûr, fondé sur l'hypothèse générale de l'existence de permutations à sens-unique à trappe. Cependant, chaque expéditeur et chaque destinataire dans ce schéma doivent faire confiance à un centre - un troisième participant qui génère et leur distribue la clef publique ainsi que la clef secrète. Par conséquent, ce schéma ne rentre pas dans la catégorie classique des schémas de chiffrement à clef publique [41].

1.1.6 Non-malléabilité

La construction d'un schéma $\mathsf{IND}-\mathsf{CCA2}$ sûr est réalisée grâce à une nouvelle notion de sécurité : la *non-malléabilité* (dénotée NM). Introduite par Dolev, Dwork et Naor [43, 44], elle est considérée comme une extension de la sécurité sémantique (une autre notion de non-malléabilité équivalente et plus simple à étudier sera ultérieurement introduite par Bellare, Desai, Pointcheval et Rogaway [7]), elle est en général plus forte que l'indistinguabilité. Bellare *et. al* [7] ont pourtant prouvé que la non-malléabilité et l'indistinguabilité sont équivalentes face aux attaques à chiffrés choisis adaptatives.

La non-malléabilité exclut les attaques qui, en possession d'un chiffré, peuvent produire un nouveau chiffré tel que les clairs correspondants sont clairement reliés sans pour autant les connaître. Dolev *et. al* [43, 44] ont construit un schéma complexe qui considère l'interaction entre plusieurs chiffrements et repose sur des preuves NIZK ainsi que des signatures à usage unique (« one-time signature » en anglais). Ce schéma est prouvé non-malléable contre des attaques à chiffrés choisis adaptatives. Leur schéma est donc le premier schéma $\mathsf{IND}-\mathsf{CCA2}$ fondé sur l'hypothèse générale de l'existence de permutations à sens-unique à trappe.

Une autre construction, beaucoup plus simple, se fondant sur la même hypothèse, est due à Sahai. Avec une nouvelle notion de preuve non-malléable NIZK [109], Sahai a montré qu'une telle preuve peut être déduite d'une preuve NIZK. Il montrera ensuite qu'avec le remplacement de la preuve NIZK σ par une preuve non-malléable NIZK σ' dans le schéma $\mathsf{IND}-\mathsf{CCA1}$ de Naor et Yung, on peut rendre ce schéma résistant aux attaques à chiffrés choisis adaptatives.

À ce jour, le schéma à clef publique le plus simple et le plus efficace (dans la catégorie classique des schémas de chiffrement à clef publique) qui garantit la sécurité sémantique contre des attaques à chiffrés choisis adaptatives (et donc $\mathsf{NM} - \mathsf{CCA2}$)est le schéma de Cramer-Shoup [37]. Il s'agit du seul schéma qui ne repose pas sur la construction complexe des preuves NIZK. Cependant, d'un point de vue théorique, ce schéma est moins attractif car il ne se fonde pas sur une hypothèse générale mais sur la difficulté d'un problème décisionnel spécifique, *i.e.* le problème Diffie-Hellman Décisionnel, ou tout du moins dans des structures de groupe [39].

1.2 Notions de sécurité pour le chiffrement symétrique

1.2.1 Fonctions pseudo-aléatoires

À la différence de la cryptographie asymétrique, dans la cryptographie symétrique, le chiffrement et le déchiffrement utilisent une même clef secrète. Par conséquent, non seulement le déchiffrement mais aussi le chiffrement sont difficiles à calculer sans la connaissance de la clef. C'est pour cette raison qu'un accès à l'oracle de chiffrement peut donner un certain avantage aux attaquants. C'est pourquoi la cryptographie symétrique considère des attaques à clairs choisis. L'exigence de sécurité est que : sans la connaissance de la clef de chiffrement, la sortie est similaire à une suite de bits aléatoires même si l'on peut tester le chiffrement sur plusieurs couples clair-chiffré. Autrement dit, la sortie est pseudo-aléatoire au sens où il n'y a pas d'algorithme faisable (*i.e.* un attaquant en temps polynomial) qui puisse la distinguer d'une suite de bits véritablement aléatoires.

1.2.2 Permutations (super) pseudo-aléatoires

Pour la catégorie de chiffrement la plus fréquemment étudiée en cryptographie symétrique — le chiffrement par bloc — la propriété désirée est que le chiffrement soit une permutation (super) pseudo-aléatoire. Une permutation pseudo-aléatoire est une fonction pseudo-aléatoire où la fonction est une permutation. La notion de permutation super pseudo-aléatoire est une extension de permutation pseudo-aléatoire où l'attaquant peut accéder non seulement à la permutation mais aussi à son inverse. Cette notion modélise la propriété la plus forte concernant les chiffrements par bloc. Ces derniers seront, à leur tour, utilisés comme primitives pour d'autres applications (chiffrement de données à travers des modes d'opération, code d'authentification de messages,...). Une question générale est de savoir s'il existe des constructions de fonctions pseudo-aléatoires ou permutations (super) pseudo-aléatoires ; et si oui, sous quelle hypothèse.

Goldreich, Goldwasser et Micali [56] ont présenté une construction simple de fonctions pseudo-aléatoires à partir d'un générateur pseudo-aléatoire de longueur double (*i.e.* de n bits à $2n$ bits). À la différence des fonctions pseudo-aléatoires, un générateur pseudo-aléatoire donne une suite de bits pseudo-aléatoires à partir d'une suite plus courte de

bits vraiment aléatoires. Blum et Micali [17, 18], en utilisant un *prédicat difficile* [1] de cette fonction, ont construit un générateur pseudo-aléatoire à partir d'une permutation à sens-unique de façon très efficace. Hastad, Impagliazzo, Levin et Luby ont montré dans [64] (qui est une combinaison des résultats de [65, 63]) qu'un générateur pseudo-aléatoire existe si et seulement si une fonction à sens-unique existe. Bien que leur construction ne soit pas efficace et n'ait jamais été utilisée en pratique, c'est un résultat important d'un point de vue théorique.

Luby et Rackoff [78] ont construit des permutations super pseudo-aléatoires à partir de fonctions pseudo-aléatoires. La construction comprend une construction de Feistel à 4 tours (3 tours pour des permutations pseudo-aléatoires). La réponse concernant l'existence des permutations super pseudo-aléatoires est ainsi affirmative sous la seule hypothèse générale de l'existence de fonctions à sens-unique.

1.3 Discussion

1.3.1 Du point de vue théorique

Reprenons la question « la cryptographie peut-elle être uniquement fondée sur les hypothèses générales ? ». Dans ce travail, nous étudions le cas du chiffrement. Nous avons vu dans les sections précédentes que, pour le chiffrement asymétrique, l'existence de permutations à sens-unique à trappe suffit pour construire des schémas $\mathsf{IND}-\mathsf{CCA2}$ sûrs et que, pour le chiffrement symétrique, l'existence de fonctions à sens-unique suffit pour construire des permutations super pseudo-aléatoires.

Les fonctions à sens-unique sont-elles suffisantes pour le chiffrement asymétrique ? Impagliazzo et Rudich [66] ont partiellement répondu négativement à cette question en prouvant que ces fonctions (voire les permutations à sens-unique comme indiqué après par [70]) ne permettent pas de construction générique d'un schéma $\mathsf{IND}-\mathsf{CPA}$ sûr. Cela signifie que l'on ne peut utiliser les fonctions à sens-unique comme une « boîte noire » pour construire des schémas de chiffrement asymétrique. Cependant, il ne s'agit pas d'une impossibilité absolue ; une construction spécifique et concrètes d'un schéma de chiffrement à clef publique pourra être faisable à partir de fonctions à sens-unique.

Alors que la condition minimale pour chiffrement symétrique est évidemment l'existence de fonctions à sens-unique, celle pour chiffrement asymétrique n'a pas encore été trouvée. Même si on se contente de la notion $\mathsf{OW}-\mathsf{CPA}$, on ne sait pas construire de schéma de chiffrement à partir de fonctions à sens-unique sans trappe. En effet, on remarque d'abord que l'existence des schémas $\mathsf{OW}-\mathsf{CPA}$ sûrs est équivalente à l'existence

[1] Un prédicat difficile B d'une fonction f est un prédicat booléen tel que $B(x)$ est efficacement calculé, étant donné x mais il est difficilement calculé, étant donné juste $f(x)$. Il y a une manière simple de modifier une fonction à sens-unique en une autre fonction à sens-unique telle qu'elle possède un prédicat difficile de la dernière [57].

d'une famille de *prédicats à trappe* [2] [59]. Bellare *et. al* [8] ont montré que les fonctions à sens-unique à trappe, qui sont non injectives et dont la taille des pré-images correspondant à une image est d'ordre polynomial, suffisent pour construire des prédicats à trappe. Dans un effort de construire ces fonctions, Bellare *et. al* ont pu obtenir, à partir des fonctions à sens-unique, des fonctions à sens-unique à trappe non injectives dont la taille des pré-images correspondant à une image est d'ordre super-polynomial. Bien qu'elles ne servent pas à construire des schéma $\mathsf{OW}-\mathsf{CPA}$ sûrs, il s'agit du premier travail ayant pour but d'éviter « la trappe » dans la construction des schéma de chiffrement asymétrique.

1.3.2 Du point de vue pratique

En pratique, on doit trouver un compromis entre l'efficacité (la complexité) du schéma et son niveau de sécurité en faisant quelques hypothèses sur l'attaque. À titre d'exemple, l'attaque pourrait être générique et indépendante de l'exécution réelle de certains objets tels que les fonctions de hachage (dans le modèle de l'oracle aléatoire [49, 9]), le chiffrement symétrique par bloc (dans le modèle du « chiffrement idéal ») ou les groupes algébriques (dans le modèle générique [27]). Parmi ces hypothèses idéalistes, celle sur les fonctions de hachage — le modèle de l'oracle aléatoire — est la plus étudiée. Dans ce « modèle de l'oracle aléatoire », la fonction de hachage est formalisée comme un oracle qui retourne une valeur parfaitement aléatoire. On a mis au point des schémas particulièrement efficaces dans ce modèle comme OAEP pour le chiffrement ou PSS (*Probabilistic Signature Scheme*) [11, 35] pour la signature.

Alors que dans le modèle standard, on ne peut être sûr de l'existence d'une transformation des schémas $\mathsf{IND}-\mathsf{CPA}$ sûrs en schémas $\mathsf{IND}-\mathsf{CCA2}$ sûrs, une telle transformation existe dans le modèle de l'oracle aléatoire [51]. De plus, dans [100], Pointcheval a montré une méthode générique très efficace pour construire des schémas $\mathsf{IND}-\mathsf{CCA2}$ sûrs à partir des schémas $\mathsf{OW}-\mathsf{CPA}$ sûrs (le même résultat avec une construction moins efficace a également été indépendamment proposé par Fujisaki et Okamoto [52, 53]). Cela explique la simplicité, et donc, l'efficacité, des schémas prouvés sûrs dans le modèle de l'oracle aléatoire.

1.4 Formalisation de quelques notions de base

Dans les analyses ultérieures, nous utilisons souvent les notions de fonctions négligeables, de famille de fonctions à sens-unique et de famille de fonctions pseudo-aléatoires

[2] Un *prédicat à trappe* (« trapdoor predicate » en anglais) est, informellement, une fonction probabiliste vers {0,1} qui ne peut être efficacement évaluée qu'avec une *trappe*. Goldwasser et Micali [59] ont montré que les prédicats à trappe sont équivalents aux schémas de chiffrement d'un bit $\mathsf{IND}-\mathsf{CPA}$ sûrs qui sont, à leur tour, équivalents aux schémas $\mathsf{IND}-\mathsf{CPA}$ sûrs par une transformation « bit par bit ». Comme le chiffrement d'un bit $\mathsf{IND}-\mathsf{CPA}$ sûr est équivalent au chiffrement d'un bit $\mathsf{OW}-\mathsf{CPA}$ sûr, on déduit que les prédicats à trappe sont équivalents aux schémas $\mathsf{OW}-\mathsf{CPA}$ sûrs.

pour le chiffrement asymétrique et le chiffrement symétrique. Nous présentons ici la définition formelle de ces notions de base :

Définition 1 (Fonction négligeable) *Une fonction $\mu : \mathbb{N} \to \mathbb{R}^+$ est dite négligeable si, pour tout polynôme $p(.)$, il existe un nombre entier N tel que $\mu(n) < \frac{1}{p(n)}$ pour tout $n \geq N$.*

Définition 2 (Famille de fonctions à sens-unique) *Étant donné une famille de fonctions $(f_i : D_i \to \{0,1\}^\star)_{i \in I}$ et deux fonctions $t : \mathbb{N} \to \mathbb{N}$ et $\varepsilon : \mathbb{N} \to \mathbb{R}^+$. On dit que la famille de fonctions $(f_i)_{i \in I}$ est (t, ε)-OW si les conditions suivantes sont satisfaites :*

Facile à échantillonner : *Il existe une machine de Turing polynomiale probabiliste Id et un polynôme $p(.)$ tels que, à partir d'une entrée 1^n, Id retourne un indice $i \in I \cap \{0,1\}^{p(n)}$. Il existe un algorithme Dom qui, à partir de l'entrée $i \in I$, retourne $x \in D_i$.*

Facile à calculer : *Il existe une machine de Turing polynomiale déterministe $Eval$ qui calcule, à partir des entrées $i \in I$ et $x \in D_i$, $f_i(x)$,* i.e. *$Eval(i, x) = f_i(x)$.*

Difficile à inverser : *Pour tout attaquant $\mathcal{A}$, on définit la probabilité de succès de $\mathcal{A}$ d'inverser la famille $(f_i)_{i \in I}$ par :*

$$\mathsf{Succ}^{\mathsf{ow}}_f(\mathcal{A}, n) \stackrel{\mathsf{def}}{=} \Pr_{i \leftarrow Id(1^n), x \leftarrow Dom(i)}[f(\mathcal{A}(i, f(x))) = f(x)].$$

On note que dans cette définition, la probabilité est prise sur i, x et sur tous les aléas dans l'algorithme $\mathcal{A}$. On définit aussi le succès maximal de tous les attaquants qui s'exécutent en un temps borné par $t(n)$ par $\mathsf{Succ}^{\mathsf{ow}}_f(t)$-une fonction de n.
Alors, pour tout n suffisamment grand, $\mathsf{Succ}^{\mathsf{ow}}_f(t)$ est borné par $\varepsilon(n)$.

$(f_i)_{i \in I}$ est également dite famille de fonctions à sens-unique si, pour tout polynôme $t(n)$, il existe une fonction négligeable $\varepsilon(n)$ telle que $(f_i)_{i \in I}$ est (t, ε)-OW.

Considérons l'exemple typique d'un candidat pour des fonctions à sens unique : les fonctions RSA. La famille de fonctions RSA $(RSA_i : D_i \to D_i)_{i \in I}$ est définie par :

- L'ensemble d'indices est composé de couples (N, e), où N est le produit de deux nombres premiers P, Q de taille n et e est un entier plus petit que N et premier avec $(P-1)(Q-1)$.
- Pour chaque $i = (N, e)$, D_i est défini par $D_i \stackrel{\mathsf{def}}{=} \mathbb{Z}^\star_N$.
- Pour tout $x \in \mathbb{Z}^\star_N$, $RSA_{(N,e)}(x) \stackrel{\mathsf{def}}{=} x^e \bmod N$

On remarque que pour tout (N, e), $RSA_{(N,e)}$ est une permutation. La famille de fonctions RSA est, en fait, une famille de permutations. On montre que cette famille satisfait les deux premières conditions d'une famille de fonctions à sens-unique (*i.e.* facile à échantillonner et facile à calculer) grâce aux trois algorithmes $(Id, Dom, Eval)$ suivant :

- L'algorithme Id, sur l'entrée 1^n, choisit uniformément deux nombres premiers P, Q de taille n (pour choisir uniformément un nombre premier de taille n, on prend aléatoirement un nombre entier de taille n et on teste s'il est premier ce on se fait en temps polynomial [1]). Id retourne $N = P.Q$ et un entier e, plus petit que N et premier avec $(P-1)(Q-1)$.

– L'algorithme Dom, sur l'entrée (N, e), choisit aléatoirement $x \in \mathbb{Z}_N^\star$.

– L'algorithme $Eval$ est défini par : $Eval((N, e), x) \stackrel{\mathsf{def}}{=} x^e \bmod N$, pour tout $x \in \mathbb{Z}_N^\star$.

En ce qui concerne la troisième condition pour être une famille de fonctions à sens-unique (*i.e.* dificile à inverser), on remarque que la meilleure méthode connu pour inverser la fonction $RSA_{(N,e)}$ est la factorisation de N. En effet, un problème ouvert bien connu est de savoir si la factorisation de N peut être réduite à une inversion de la fonction $RSA_{(N,e)}$. De plus, lemeilleur algorithme connu pour la factorisation n'est pas polynomial.

Définition 3 (Famille de permutations à sens-unique à trappe) *Étant donné une famille de permutations $(f_i : D_i \to D_i)_{i \in I}$ et deux fonctions $t : \mathbb{N} \to \mathbb{N}$ et $\varepsilon : \mathbb{N} \to \mathbb{R}^+$. On dit que la famille de permutations $(f_i)_{i \in I}$ est (t, ε)-OW à trappe si les conditions suivantes sont satisfaites :*

Facile à échantillonner : *Il existe une machine de Turing polynomiale probabiliste Id et deux polynômes $p(.), q(.)$ tels que, à partir d'une entrée 1^n, Id retourne un indice $i \in I \cap \{0,1\}^{p(n)}$ et une trappe $\mathsf{td} \in \{0,1\}^{q(n)}$. Il existe un algorithme Dom qui, sur l'entrée $i \in I$, retourne $x \in D_i$.*

Facile à calculer : *Il existe une machine de Turing polynomiale déterministe $Eval$ qui calcule, à partir des entrées $i \in I$ et $x \in D_i$, $f_i(x)$,* i.e. *$Eval(I, x) = f_i(x)$.*

Facile à inverser avec trappe : *il existe une machine de Turing polynomiale déterministe $Eval^{-1}$ qui calcule, pour tout $(i, \mathsf{td}) \leftarrow I$ et pour tout $x \in D_i$, $f_i^{-1}(f_i(x))$,* i.e. *$Eval^{-1}(\mathsf{td}, f_i(x)) = x$.*

Difficile à inverser sans trappe : *Pour tout attaquant $\mathcal{A}$, on définit la probabilité de succès de $\mathcal{A}$ d'inverser la famille $(f_i)_{i \in I}$ par :*

$$\mathsf{Succ}_f^{\mathsf{ow}}(\mathcal{A}, n) \stackrel{\mathsf{def}}{=} \Pr_{i \leftarrow Id(1^n), x \leftarrow Dom(i)}[f(\mathcal{A}(i, f(x))) = f(x)].$$

On note que dans cette définition, la probabilité est prise sur i, x et sur tous les aléas dans l'algorithme $\mathcal{A}$. On définit aussi le succès maximal de tous les attaquants qui s'exécutent en un temps borné par $t(n)$ par : $\mathsf{Succ}_f^{\mathsf{ow}}(t)$-une fonction de n.
Alors, pour tout n suffisamment grand, $\mathsf{Succ}_f^{\mathsf{ow}}(t)$ est bornée par $\varepsilon(n)$.

On dit aussi que $(f_i)_{i \in I}$ est une famille de permutations à sens-unique à trappe si, pour tout polynôme $t(n)$, il existe une fonction négligeable $\varepsilon(n)$ telle que $(f_i)_{i \in I}$ soit (t, ε)-OW à trappe.

On voit que la famille de permutations RSA présentée ci-dessus peut être modifiée pour avoir la propriété de trappe : l'algorithme Id est simplement modifié pour retourner l'indice (N, e) et la trappe (N, d) où $d = e^{-1} \bmod (P-1)(Q-1)$. L'algorithme d'inversion $Eval^{-1}$ est défini par $Eval^{-1}((N, d), y) = y^d \bmod N$.

Définition 4 (Famille de fonctions pseudo-aléatoires) *Soit une famille de fonctions $F : Keys(F) \times D \to R$.*
On appelle $U^{D \to R}$ l'ensemble de toutes les fonctions de $D \to R$.
Considérons un attaquant $\mathcal{A}$ qui a accès à un oracle $\mathcal{O}_b$, où :

- $\mathcal{O}_0$ *est une fonction* g *aléatoirement choisie dans* $U^{D\to R}$ *:* $g \xleftarrow{R} U^{D\to R}$*, on dit que* $\mathcal{O}_0$ *est une fonction véritablement aléatoire.*
- $\mathcal{O}_1$ *est une fonction* g *aléatoirement choisie dans la famille* F *:* $g \xleftarrow{R} F$*. Autrement dit,* $k \xleftarrow{R} Keys(F)$ *et* $g \stackrel{\mathsf{def}}{=} F_k$*.*

L'avantage de l'attaquant $\mathcal{A}$ *pour deviner le bit* b *est défini par :*

$$\mathsf{Adv}_F^{\mathsf{prf}}(\mathcal{A}) = 2 \times \Pr_{b,g\xleftarrow{R}U^{D\to R},k\xleftarrow{R}Keys(F)}[\mathcal{O}_0 = g, \mathcal{O}_1 = F_k, \mathcal{A}^{\mathcal{O}_b} = b] - 1.$$

On définit aussi $\mathsf{Adv}_F^{\mathsf{prf}}(t,n)$*, l'avantage maximal de tous les attaquants qui s'exécutent en un temps borné par* t *et qui font au plus* n *requêtes à l'oracle* $\mathcal{O}_b$*.*
On dit que la famille F *est* (ε,t,n)-PRF *si* $\mathsf{Adv}_F^{\mathsf{prf}}(t,n)$ *est borné par* ε*.*

Discussion sur le temps de fonctionnement d'une attaque Dans les définitions ci-dessus et dans celles qui suivent dans cette thèse, le temps d'exécution d'une attaque englobe le temps de calcul et le temps de chargement (ou taille) du code de l'algorithme d'attaque. Ainsi, si les attaques considérées fonctionnent en temps polynomial, le code de l'algorithme a nécessairement une taille polynomiale.

Notons que sans cette convention, certaines hypothèses deviendraient inatteignables. Considérons par exemple un schéma de chiffrement par blocs E_k. Si on ne tient pas compte de la taille du code, le code peut contenir toutes les valeurs de $(E_k(m), m, k)$, pour toutes les clés et tous les clairs. Lors de l'attaque, sur un couple texte clair - texte chiffré, une consultation de la table permet de retrouver la clé k. La résistance aux attaques à clairs-chiffrés connus serait impossible.

2

Analyse des notions de sécurité pour le chiffrement asymétrique

Sommaire

Dans les applications réelles, le contexte est étudié afin de choisir le niveau de sécurité adéquat. Puisqu'il n'est pas toujours nécessaire d'utiliser le schéma le plus sûr, il est important d'étudier non seulement les méthodes de construction des schémas mais aussi la relation entre les différentes notions de sécurité. À titre d'exemple, lorsqu'on utilise un schéma de chiffrement asymétrique sémantiquement sûr contre les attaques à chiffrés choisis non-adaptatives, est-il nécessairement non-malléable en considérant seulement des attaques passives ? La réponse est négative [7]. Dans le contexte où la non-malléabilité est importante, un tel schéma (qui est apparemment très sûr) ne suffit pas. La recherche sur les relations entre les notions de sécurité devient très importante : un schéma sûr dans un sens est-il sûr dans un autre sens ?

Les relations entre les notions de sécurité ont été étudiées de manière approfondie par Bellare, Desai, Pointcheval et Rogaway [7], puis par Bellare et Sahai [12]. Dans [7], les auteurs ont montré les séparations et les implications entre les principales notions de sécurité dans le cadre du chiffrement asymétrique. Le résultat est résumé dans la figure 2.1

Dans [12], Bellare et Sahai ont présenté la notion d'attaques parallèles (dénotées PA,

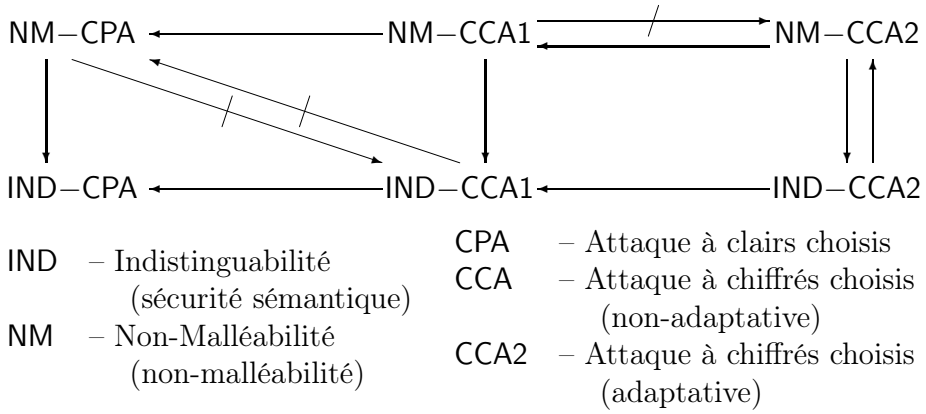

FIG. 2.1 – Relations entre les notions de sécurité

par *parallel attack* en anglais). Dans ce modèle, l'attaquant peut soumettre à la fin (avant de retourner la réponse) un vecteur de chiffrés à l'oracle de déchiffrement. Les auteurs ont montré que la notion NM selon des attaques normales est équivalente à la notion IND selon les attaque parallèles.

Dans notre travail [99], nous considérons les cas concrets en introduisant des niveaux de sécurité $(i, j) - \mathsf{IND}$, $(i, j) - \mathsf{NM}$ où l'attaquant peut poser au maximum i requêtes avant et j requêtes après la réception du challenge. La raison d'une telle notation, plus précise que $\mathsf{IND}-\mathsf{CCA1}$ capturée par $(poly(.), 0)-\mathsf{IND}$ (où $poly(.)$ désigne un polynôme)et $\mathsf{IND}-\mathsf{CCA2}$ capturée par $(poly(.), poly(.))-\mathsf{IND}$, est que l'on peut prouver l'importance de chaque requête : une requête avant de recevoir le challenge ne peut pas être remplacée par une, voire plusieurs requêtes, après avoir reçu le challenge et inversement. Cela contredit l'intuition qu'une requête après réception du challenge est plus importante qu'une requête avant la réception du challenge. Cette notation précise nous permet aussi de mieux appréhender la relation entre l'indistinguabilité et la non-malléabilité.

Comme application, nous introduisons le nouveau modèle d'attaque « à chiffrés choisis post-challenge » (dénoté CCAO2) où l'attaquant ne peut poser des requêtes de déchiffrement qu'après avoir reçu le challenge. Nous montrons que cette notion est indépendante des autres modèles : CCA1 et CCAO2 sont indépendantes et $\mathsf{CCA1} + \mathsf{CCAO2}$ n'implique pas CCA2. D'un point de vue pratique, ce genre d'attaques modélise un scénario très réaliste où l'attaquant commence son attaque à partir du moment où il est au courant de l'importance d'un chiffré spécifique (après que ce dernier a été fabriqué).

2.1 Notions de sécurité pour le chiffrement asymétrique

On rappelle d'abord la notion formelle de schéma de chiffrement asymétrique :

2.1.1 Schéma de chiffrement asymétrique

Un schéma de chiffrement asymétrique π est défini par les trois algorithmes suivants :
- L'*algorithme de génération de clef* $\mathcal{K}$. Sur l'entrée 1^k, où k est le paramètre de sécurité, l'algorithme $\mathcal{K}$ produit un couple $(\mathsf{pk}, \mathsf{sk})$ constitué d'une clef publique et d'une clef secrète. On dénote $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k)$
- L'*algorithme de chiffrement* $\mathcal{E}$. Étant donné un message m (dans l'espace des textes clairs $\mathcal{M}$) et une clef publique pk, $\mathcal{E}_{\mathsf{pk}}(m)$ produit un texte chiffré c (dans l'espace des chiffrés $\mathcal{C}$) de m. Cet algorithme peut être probabiliste et nécessite alors un aléa supplémentaire $r \in \mathcal{R}$; il est dénoté par $\mathcal{E}_{\mathsf{pk}}(m; r)$.
- L'*algorithme de déchiffrement* $\mathcal{D}$. Étant donné un texte chiffré $c \in \mathcal{C}$ et une clef secrète sk, $\mathcal{D}_{\mathsf{sk}}(c)$ retourne un texte clair $m \in \mathcal{M}$ ou un symbole invalide $\perp$. C'est typiquement un algorithme déterministe.

La condition de consistance requise est que le déchiffrement d'un chiffrement redonne le texte clair original : pour tout couple $(\mathsf{pk}, \mathsf{sk})$ généré par l'algorithme de génération de clef, $\mathcal{D}_{\mathsf{sk}}(\mathcal{E}_{\mathsf{pk}}(m; r)) = m$ pour tout $m \in \mathcal{M}$ et tout $r \in \mathcal{R}$.

Nous rappelons ensuite les formalisations de différentes notions de sécurité qui ont été informellement présentées dans l'introduction.

2.1.2 Niveau de sécurité

Définition 5 (Sens-unique — One-wayness) *Soit un schéma de chiffrement $\pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. Considérons un attaquant $\mathcal{A}$. On définit le succès de $\mathcal{A}$ d'inverser le chiffrement du schéma π par :*

$$\mathsf{Succ}^{\mathsf{ow}}_{\pi}(\mathcal{A}) = \Pr_{m,r}\left[(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k) : \mathcal{A}(\mathsf{pk}, \mathcal{E}_{\mathsf{pk}}(m, r)) = m\right].$$

On définit aussi le succès maximal sur tous les attaquants $\mathcal{A}$, qui fonctionnent en temps borné par t, par $\mathsf{Succ}^{\mathsf{ow}}_{\pi}(t)$. Alors, on dit que π est $(t, \varepsilon)-\mathsf{OW}$ sûr si $\mathsf{Succ}^{\mathsf{ow}}_{\pi}(t)$ est plus petit que ε.

Définition 6 (Indistinguabilité) *Soit un schéma de chiffrement $\pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. Considérons une attaque à deux étapes $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. On définit l'avantage de $\mathcal{A}$ contre le schéma $\pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ au sens de l'indistinguabilité par :*

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{A}) &= \left| 2 \times \Pr_{b,r}\left[\begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k), (m_0, m_1, s) \leftarrow A_1(\mathsf{pk}), \\ c = \mathcal{E}_{\mathsf{pk}}(m_b, r), b' = A_2(m_0, m_1, s, c) : b' = b\end{array}\right] - 1 \right| \\
&= \left| \Pr_{r}[b' = 1 \,|\, b = 1] - \Pr_{r}[b' = 1 \,|\, b = 0] \right|.
\end{aligned}$$

On insiste sur le fait que $\mathcal{A}_1$ retourne deux messages m_0 et m_1 de même longueur $|m_0| = |m_1|$. On définit aussi l'avantage maximal sur tous les attaquants $\mathcal{A}$, qui fonctionnent en temps borné par t, par $\mathsf{Adv}^{\mathsf{ind}}_{\pi}(t)$. Alors, on dit que π est $(t, \varepsilon)-\mathsf{IND}$ sûr si $\mathsf{Adv}^{\mathsf{ind}}_{\pi}(t)$ est plus petit que ε.

Définition 7 (Non-malléabilité) *Soit un schéma de chiffrement $\pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$. Considérons une attaque à deux étapes $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. On définit l'avantage de $\mathcal{A}$ contre le schéma $\pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ au sens de la non-malléabilité par :*

$$\mathsf{Adv}^{\mathsf{nm}}_{\pi}(\mathcal{A}) = \left|\mathsf{Succ}^{M}_{\pi}(\mathcal{A}) - \mathsf{Succ}^{\$}_{\pi}(\mathcal{A})\right|, \text{ avec}$$

$$\left.\begin{array}{rcl} \mathsf{Succ}^{M}_{\pi}(\mathcal{A}) & = & \Pr\left[\begin{array}{c} y \notin \mathbf{y} \ \wedge \ \perp \notin \mathbf{x} \\ \wedge \ R(x, \mathbf{x}) \end{array}\right] \\ \mathsf{Succ}^{\$}_{\pi}(\mathcal{A}) & = & \Pr\left[\begin{array}{c} y \notin \mathbf{y} \ \wedge \ \perp \notin \mathbf{x} \\ \wedge \ R(x^\star, \mathbf{x}) \end{array}\right] \end{array}\right\} \begin{array}{c} \text{sur l'espace de probabilités défini par} \\ \\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k), (M, s) \leftarrow A_1(\mathsf{pk}), \\ x, x^\star \leftarrow M, y = \mathcal{E}_{\mathsf{pk}}(x, r), \\ (R, \mathbf{y}) \leftarrow A_2(M, s, y), \mathbf{x} = \mathcal{D}_{\mathsf{sk}}(\mathbf{y}). \end{array}$$

On définit aussi l'avantage maximal sur tous les attaquants $\mathcal{A}$, qui fonctionnent en temps borné par t, par $\mathsf{Adv}^{\mathsf{nm}}_{\pi}(t)$. Alors, on dit que π est $(t, \varepsilon)-\mathsf{NM}$ sûr si $\mathsf{Adv}^{\mathsf{nm}}_{\pi}(t)$ est plus petit que ε.

2.1.3 Puissance de l'attaquant

Définition 8 (Attaque à chiffrés choisis non-adaptative) *Un attaquant est appelé* à chiffrés choisis non-adaptatif*,(dénoté* CCA1*) s'il ne peut accéder à l'oracle de déchiffrement après avoir eu connaissance du challenge.*

Définition 9 (Attaque à chiffrés choisis adaptative) *Un attaquant est appelé* à chiffrés choisis adaptatif*, (dénoté* CCA2*) s'il peut accéder à l'oracle de déchiffrement à tout instant,* i.e. *avant et après la réception du challenge ; la seule restriction étant de ne pas faire de requête sur le challenge lui-même.*

Pour plus de généralité, nous présentons une définition plus précise avec un (i, j)-*attaquant* qui peut faire au plus i requêtes (resp. j requêtes) avant la réception du challenge (resp. après la réception du challenge)

Définition 10 (Attaque à chiffrés choisis) *Un attaquant est un* (i, j)-attaquant à chiffrés choisis*, (dénoté $(i, j)-\mathsf{CCA}$) s'il peut faire au maximum i requêtes (resp. j requêtes) à l'oracle avant (resp. après) la réception du challenge ; la seule restriction étant de ne pas faire de requête sur le challenge lui-même.*

Notation. Un schéma de chiffrement $\pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ est dit $(t, \varepsilon)-\mathsf{XXX}-\mathsf{YYY}$ sûr si pour tout YYY-attaquant $\mathcal{A}$ au sens de la sécurité XXX en temps t, l'avantage de $\mathcal{A}$ est borné par ε. Ici, XXX peut être IND ou NM, et YYY peut être CPA, $\mathsf{CCA1}$, $\mathsf{CCA2}$, ou $(i, j)-\mathsf{CCA}$. Pour simplifier la notation, on dit aussi que π est $(t, \varepsilon, i, j)-\mathsf{IND}$ sûr (resp. $(t, \varepsilon, i, j)-\mathsf{NM}$ sûr) si pour tout $(i, j)-\mathsf{CCA}$ attaquant $\mathcal{A}$ dont le temps de calcul est borné par t, $\mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{A}) \leq \varepsilon$ (resp. $\mathsf{Adv}^{\mathsf{nm}}_{\pi}(\mathcal{A}) \leq \varepsilon$.)

2.2 Relations concrètes entre les notions de sécurité

Dans cette partie, les relations concrètes entre les notions de sécurité seront démontrées. On verra ensuite que les preuves des résultats précédents [7] peuvent être déduites de notre théorème 15.

Nous allons montrer quelques différences non-intuitives dans les classes $(i, j)-\mathsf{IND}$: une requête de déchiffrement dans la première étape (avant la connaissance du challenge) ne peut pas être retardée à la deuxième étape (après connaissance du challenge) et inversement. De manière formelle, nous prouverons qu'une requête de plus dans la première étape peut donner plus d'avantage à l'attaquant que plusieurs requêtes dans la deuxième étape. L'inverse est aussi vrai, *i.e.* une requête dans la deuxième étape ne peut pas être remplacée par plusieurs requêtes dans la première étape.

Pour cela, nous introduisons un nouveau problème calculatoire lié aux fonctions pseudo-aléatoires.

Définition 11 *Étant donné une fonction G, aléatoirement choisie dans une famille $\mathcal{G}$ de fonctions/permutations de E dans E, et une attaque à deux étapes $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ où $\mathcal{A}_1$ et $\mathcal{A}_2$ ne peuvent respectivement faire qu'au plus m et $n-1$ requêtes à G. Nous définissons $\mathsf{Succ}^{m,n}_{\mathcal{G}}(\mathcal{A})$ la probabilité pour $\mathcal{A}$ de calculer $G^n(v)$ sur une instance aléatoire $v \in E$:*

$$\mathsf{Succ}^{m,n}_{\mathcal{G}}(\mathcal{A}) = \Pr[G \xleftarrow{R} \mathcal{G}; v \xleftarrow{R} E; s \leftarrow \mathcal{A}_1^G : \mathcal{A}_2^G(v, s) = G^n(v)].$$

Nous dénotons aussi $\mathsf{Succ}^{m,n}_{\mathcal{G}}(t)$ la valeur maximale de $\mathsf{Succ}^{m,n}_{\mathcal{G}}(\mathcal{A})$ sur tous les attaquants dont le temps de calcul est borné par t.

Proposition 12 *Soit $\mathcal{G}$ une famille de fonctions/permutations pseudo-aléatoires de E dans E. Pour tout ℓ tel que le cardinal de E soit supérieur à 2^ℓ, on a :*

$$\mathsf{Succ}^{m,n}_{\mathcal{G}}(t) \leq \mathsf{Adv}^{\mathsf{prf}}_{\mathcal{G}}(m + 2n - 1, t) + \frac{mn+1}{2^\ell}.$$

Preuve. Considérons un attaquant $\mathcal{A}$ contre la famille $\mathcal{G}$ dont le succès $\mathsf{Succ}^{m,n}_{\mathcal{G}}(t)$ est non-négligeable. Nous construisons un attaquant-PRF $\mathcal{B}$ tel que $\mathsf{Succ}^{m,n}_{\mathcal{G}}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{prf}}_{\mathcal{G}}(\mathcal{B})$. L'attaquant $\mathcal{B}$ fonctionne de la manière suivante : quand $\mathcal{A}$ fait une requête à G, $\mathcal{B}$ fait la même requête à $\mathcal{O}_b$ et transmet la réponse à $\mathcal{A}$ (au plus $m + n - 1$ requêtes au total.) À la fin du processus, $\mathcal{A}$ retourne x. Alors, $\mathcal{B}$ fait successivement des requêtes à l'oracle $\mathcal{O}_b$ pour obtenir $y = \mathcal{O}_b^n(v)$. Si $x = y$, $\mathcal{B}$ retourne le bit $b' = 1$, sinon $\mathcal{B}$ retourne le bit $b' = 0$.

– si $b = 1$, $\mathcal{B}$ a accès à G et $y = \mathcal{O}_b^n(v) = G^n(v)$. $\mathcal{B}$ gagne toujours le jeu si $\mathcal{A}$ gagne : $b' = 1$ signifie que $x = y$.

$$\mathsf{Adv}^{\mathsf{prf}}_{\mathcal{G}}(\mathcal{B} \,|\, b = 1) = 2\Pr[x = y \,|\, b = 1] - 1 = 2\mathsf{Succ}^{m,n}_{\mathcal{G}}(\mathcal{A}) - 1.$$

– si $b = 0$, $y = \mathcal{O}_b^n(v)$ est parfaitement aléatoire et indépendant de la vue de $\mathcal{A}$, sauf si $\mathcal{A}_1$ a fait une requête sur $\mathcal{O}_b^i(v)$ (pour $0 \leq i < n$). On obtient donc :

$$\mathsf{Adv}_{\mathcal{G}}^{\mathsf{prp}}(\mathcal{B} \mid b = 0) = 2\Pr[x = y \mid b = 0] - 1 \leq 2 \times \left(\frac{mn}{2^\ell} + \frac{1}{2^\ell}\right) - 1.$$

En combinant les deux cas, en sachant que b est un bit aléatoire, on obtient le résultat. □

La construction et la proposition qui suivent seront utilisées plusieurs fois dans nos preuves de sécurité.

Définition 13 *Soient $\pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ un schéma de chiffrement public et f une permutation sur $\mathcal{M}$, l'inverse de f est dénoté par f^{-1}. Nous définissons un nouveau schéma de chiffrement $\pi^{(f)} = (\mathcal{K}^{(f)}, \mathcal{E}^{(f)}, \mathcal{D}^{(f)})$:*

$$\mathcal{M}^{(f)} = \mathcal{M} \qquad \mathcal{R}^{(f)} = \mathcal{R} \qquad \mathcal{C}^{(f)} = \mathcal{C}$$

Algorithme $\mathcal{K}^{(f)}(1^k)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k)$
$\mathsf{pk}^{(f)} \leftarrow \mathsf{pk}||f||f^{-1}$
$\mathsf{sk}^{(f)} \leftarrow \mathsf{sk}$
retourner $(\mathsf{pk}^{(f)}, \mathsf{sk}^{(f)})$

Algorithme $\mathcal{E}^{(f)}_{\mathsf{pk}^{(f)}}(m, r)$
$\mathsf{pk}||f||f^{-1} \stackrel{\mathsf{def}}{=} \mathsf{pk}^{(f)}$
retourner $\mathcal{E}_{\mathsf{pk}}(f(m), r)$

Algorithme $\mathcal{D}^{(f)}_{\mathsf{sk}^{(f)}}(c)$
$\mathsf{sk} \stackrel{\mathsf{def}}{=} \mathsf{sk}^{(f)}$
retourner $f^{-1}(\mathcal{D}_{\mathsf{sk}}(c))$

Proposition 14 *Soient $\pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ un schéma de chiffrement et f une permutation, où f et f^{-1} sont efficacement calculables, π et $\pi^{(f)}$ ont le même niveau de sécurité au sens de l'indistinguabilité quel que soit le modèle d'attaque :*

$$\mathsf{Adv}_{\pi}^{\mathsf{ind-yyy}}(t) \leq \mathsf{Adv}_{\pi^{(f)}}^{\mathsf{ind-yyy}}(t + 2T_f + q_d T_{f^{-1}}) \leq \mathsf{Adv}_{\pi}^{\mathsf{ind-yyy}}(t + (2 + q_d)(T_f + T_{f^{-1}})),$$

où T_f ($T_{f^{-1}}$ resp.) est la borne supérieure du le temps nécessaire pour évaluer f (f^{-1} resp.)

Preuve. Nous montrons d'abord que si $\pi^{(f)}$ est sûr, alors π l'est aussi. Rappelons que f et f^{-1} sont efficacement calculables et inclus dans la clef publique. Considérons un attaquant $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ contre π, nous construisons un attaquant $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ contre $\pi^{(f)}$: à chaque fois que $\mathcal{A}$ fait une requête de déchiffrement c à l'oracle $\mathcal{D}_{\mathsf{sk}}$, $\mathcal{B}$ fait la même requête à l'oracle de déchiffrement $\mathcal{D}^{(f)}_{\mathsf{sk}^{(f)}}$. $\mathcal{B}$ reçoit la réponse m et transmet $f(m)$ à $\mathcal{A}$. Quand $\mathcal{A}_1$ retourne deux candidats m_0 et m_1, $\mathcal{B}_1$ calcule $f^{-1}(m_0)$ et $f^{-1}(m_1)$. Finalement, quand $\mathcal{A}$ devine b', $\mathcal{B}$ retourne aussi cette valeur. Il est clair que l'avantage de $\mathcal{B}$ est le même que celui de $\mathcal{A}$, mais il a besoin d'évaluations additionnelles : deux pour calculer f et q_d pour calculer f^{-1}, où q_d est le nombre de requêtes à l'oracle de déchiffrement :

$$\mathsf{Adv}_{\pi}^{\mathsf{ind-yyy}}(t) \leq \mathsf{Adv}_{\pi^{(f)}}^{\mathsf{ind-yyy}}(t + 2T_f + q_d T_{f^{-1}}).$$

Puisque $\pi = \pi^{(f)(f^{-1})}$, et que f et f^{-1} sont publiques et efficacement calculables, on conclut facilement. □

Théorème 15 *Sous l'hypothèse de l'existence de permutations à sens-unique à trappe, pour tout couple* (m, n)*, il existe un schéma de chiffrement qui est* $(m, N)-\mathsf{IND}$ *et* $(M, n)-\mathsf{IND}$ *sûr mais pas* $(m+1, n+1)-\mathsf{IND}$ *sûr quels que soient* M *et* N*.*

Idée de la preuve. L'hypothèse de l'existence de permutations à sens-unique à trappe est équivalente à l'hypothèse de l'existence de schémas de chiffrement $\mathsf{IND}-\mathsf{CCA2}$-sûrs [43, 44]. Alors, on suppose qu'il existe un schéma $\pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ qui soit $\mathsf{IND}-\mathsf{CCA2}$ sûr. Il est, par conséquent, $(i, j)-\mathsf{IND}$ sûr pour tout (i, j). Supposons aussi que f soit une permutation à sens unique à trappe vers $\mathcal{M}$. D'après la proposition 14, le schéma $\pi^{(f)}$ est aussi $\mathsf{IND}-\mathsf{CCA2}$ sûr quand la trappe pour calculer f^{-1} est incluse dans la clef publique. Nous transformons $\pi^{(f)}$ en un nouveau schéma de chiffrement $\pi' = (\mathcal{K}', \mathcal{E}', \mathcal{D}')$ qui n'est plus $(m+1, n+1)-\mathsf{IND}$ sûr, mais reste à la fois $(m, N)-\mathsf{IND}$ sûr et $(M, n)-\mathsf{IND}$ sûr. Notons que dans π', la trappe pour calculer f^{-1} est désormais incluse dans la clef secrète. Ce schéma fonctionne comme suit :

- On dénote I_M un élément spécifique de $\mathcal{M}$ et pose $p_M = f^{-1}(I_M)$.
- On sélectionne deux familles : l'une de fonctions pseudo-aléatoires $\mathcal{F} = \{F_K : K \in \{0,1\}^k\}$ de $\mathcal{C}$ vers $\mathcal{C}$ et l'autre de permutations pseudo-aléatoires $\mathcal{G} = \{G_K : K \in \{0,1\}^k\}$ de $\mathcal{C}$ vers $\mathcal{C}$. On suppose, de plus, que le cardinal de $\mathcal{C}$ est plus grand que 2^ℓ.

Algorithme $\mathcal{K}'(1^k)$
$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k)$
$I_M \overset{R}{\leftarrow} \mathcal{M}, K_f, K_g \overset{R}{\leftarrow} \{0,1\}^k$
$\mathsf{pk}' \leftarrow \mathsf{pk}||f||I_M||m||n$
$\mathsf{sk}' \leftarrow \mathsf{sk}||f^{-1}||K_f||K_g$
retourner $(\mathsf{pk}', \mathsf{sk}')$

Algorithme $\mathcal{E}'_{\mathsf{pk}'}(\mu, r)$
$\mathsf{pk}||f||I_M||m||n \overset{\mathsf{def}}{=} \mathsf{pk}'$
$\varphi \leftarrow f(\mu)$
retourner $0||\mathcal{E}_{\mathsf{pk}}(\varphi, r)||\epsilon$

Algorithme $\mathcal{D}'_{\mathsf{sk}'}(b||c||z)$
$\mathsf{sk}||f^{-1}||K_f||K_g \overset{\mathsf{def}}{=} \mathsf{sk}'$
1. si $(b = 0 \wedge z = \epsilon)$ retourner $f^{-1}(\mathcal{D}_{\mathsf{sk}}(c))$
2. si $(b = 1 \wedge z = \epsilon)$ retourner $F_{K_f}(c)$
3. si $(b = 2 \wedge z = \epsilon)$ retourner $G_{K_g}(c)$
4. si $(b = 1 \wedge z = F^n_{K_f}(c) \wedge \mathcal{D}(c) = G^m_{K_g}(I_M))$ retourner $f^{-1}(G^m_{K_g}(I_M))$

sinon, retourner $\perp$

La construction est fondée sur l'idée que $m+1$ requêtes de déchiffrement dans la première étape permettent de déterminer un texte clair spécifique $\mu = f^{-1}(G^m_{K_g}(I_M))$. La spécificité vient du fait que, en faisant $n+1$ requêtes de déchiffrement dans la deuxième étape, il est possible de vérifier si un texte chiffré $0\|c\|\epsilon$ est un chiffrement de μ : d'abord on calcule $F^n_{K_f}(c)$ par n requêtes de déchiffrement, puis soumet $1\|c\|F^n_{K_f}(c)$ à l'oracle de

déchiffrement. Alors, on peut facilement construire un $(n+1, m+1)-\mathsf{CCA}$ attaquant qui casse le schéma. De plus, avec une preuve similaire à celle de la proposition 14, on peut montrer que le schéma est sûr face aux $(n, M)-\mathsf{CCA}$ attaquants et $(N, m)-\mathsf{CCA}$ attaquants.

2.2.1 Discussion sur la non-malléabilité

Nous discutons brièvement la notion « générale » de non-malléabilité (dénotée $\mathsf{CNM}^{(k)}$) dans laquelle l'attaquant produit, à la fin, un vecteur de chiffrés de dimension k, au lieu d'un seul chiffré. Dans [12], Bellare et Sahai ont présenté la notion d'attaques parallèles, dénotées PA (plus précisément $\mathsf{PA}^{(k)}$ dans notre contexte). Dans les $\mathsf{PA}^{(k)}$, après la dernière requête à l'oracle de déchiffrement, l'attaquant peut faire une requête sur un vecteur de chiffrés de taille k. Les $\mathsf{PA}^{(k)}$ peuvent être divisées en trois catégories : $\mathsf{PA0}$, $\mathsf{PA1}$ et $\mathsf{PA2}$, selon la possibilité d'accès à l'oracle de déchiffrement et sans tenir compte du vecteur de chiffrés supplémentaire (jamais, avant, avant et après la réception du challenge). Bellare et Sahai ont montré que $\mathsf{IND}-\mathsf{PAX}$ est équivalente à $\mathsf{CNM}^{(k)}-\mathsf{CCAX}$, où $\mathsf{CCA0}$ est en fait CPA. Leur résultat peut être traduit dans notre formalisation par le théorème suivant.

Théorème 16 *Les notions $(m, n)-\mathsf{IND}-\mathsf{PA}^{(k)}$ et $(m, n)-\mathsf{CNM}^{(k)}$ sont équivalentes. En d'autres termes, quel que soit le schéma de chiffrement $\pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$:*

$$\frac{1}{2} \times \mathsf{Adv}_{\pi}^{(m,n)-\mathsf{ind}-\mathsf{pa}^{(k)}}(t) \leq \mathsf{Adv}_{\pi}^{(m,n)-\mathsf{cnm}^{(k)}}(t) \leq \mathsf{Adv}_{\pi}^{(m,n)-\mathsf{ind}-\mathsf{pa}^{(k)}}(t + T_R),$$

où T_R est une borne supérieure sur temps de calcul de la relation R.

Preuve. Considérons un $(m, n)-\mathsf{IND}-\mathsf{PA}^{(k)}$-attaquant $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}'_2)$ contre π. Il s'agit d'une attaque-IND classique à deux étapes. La deuxième étape est décomposée en deux parties : $\mathcal{A}_2$ a accès à l'oracle de déchiffrement et retourne un vecteur de chiffrés ; $\mathcal{A}'_2$ reçoit le vecteur de textes clairs correspondants et retourne le bit d'estimation sans avoir besoin de l'accès à l'oracle de déchiffrement. Nous construisons un $(m, n)-\mathsf{CNM}^{(k)}$-attaquant $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ de la manière suivante :

- $\mathcal{B}_1$ exécute $\mathcal{A}_1$: à chaque fois que $\mathcal{A}_1$ fait une requête de déchiffrement, $\mathcal{B}_1$ fait la même requête à l'oracle de déchiffrement, puis transmet la réponse à $\mathcal{A}_1$. Quand $\mathcal{A}_1$ retourne deux candidats (m_0, m_1) et s, on définit et retourne une distribution uniforme $\mathcal{M} = \{m_0, m_1\}$ et l'information s ;
- Le challengeur choisit aléatoirement m suivant la distribution $\mathcal{M}$. De manière équivalente, il s'agit de choisir $b \xleftarrow{R} \{0, 1\}$ et poser $m = m_b$. Ensuite, on fabrique le challenge $c = \mathcal{E}_{\mathsf{pk}}(m, r)$ où $r \xleftarrow{R} R$;
- $\mathcal{B}_2$ reçoit l'information s et le challenge c. Il les transmet à $\mathcal{A}_2$. $\mathcal{B}_2$ exécute $\mathcal{A}_2$: à chaque fois que $\mathcal{A}_2$ fait une requête de déchiffrement, $\mathcal{B}_2$ fait la même requête à l'oracle de déchiffrement, puis transmet la réponse à $\mathcal{A}_2$. Quand $\mathcal{A}_2$ retourne le vecteur de textes chiffrés $\mathbf{y}$ et l'information s', $\mathcal{B}_2$ retourne $(R, \mathbf{y})$ où la relation R est définie par : $R(\mathbf{x}, m)$ retourne $(m = m_0) \oplus A'_2(\mathbf{x}, s')$.

Par définition, si on considère $\tilde{m}$ est défini selon $\mathcal{M}$, ce qui est équivalent à choisir aléatoirement un bit d indépendant de b et b', puis poser $\tilde{m} = m_d$, alors $\mathsf{Adv}_{\pi}^{(m,n)-\mathsf{ind}-\mathsf{cnm}^{(k)}}(\mathcal{A})$ est égal à :

$$\begin{aligned}
& \Pr[R(\mathbf{x}, m)] - \Pr[R(\mathbf{x}, \tilde{m})] = \Pr[R(\mathbf{x}, m_b)] - \Pr[R(\mathbf{x}, m_d)] \\
= \;& \frac{1}{2} \times \Big(\Pr[R(\mathbf{x}, m_b)] - \Pr[R(\mathbf{x}, m_{\bar{b}})] \Big) \\
= \;& \frac{1}{2} \times \Big(\Pr[(m_0 = m_b) \oplus A'_2(\mathbf{x}, s')] - \Pr[(m_0 = m_{\bar{b}}) \oplus A'_2(\mathbf{x}, s')] \Big) \\
= \;& \frac{1}{2} \times \Big(\Pr[(b = 0) \oplus (b' = 1)] - \Pr[(b = 0) \oplus (b' = 1)] \Big) \\
= \;& \frac{1}{2} \times \Big(\Pr[b = b'] - \Pr[b \neq b'] \Big) = \frac{1}{2} \times \mathsf{Adv}_{\pi}^{(m,n)-\mathsf{ind}-\mathsf{pa}^{(k)}}(\mathcal{B}).
\end{aligned}$$

Remarquons que le temps de calcul de $\mathcal{B}$ est exactement égal à celui de $\mathcal{A}$. On passe maintenant à la relation de droite du théorème. Considérons un $(m,n)-\mathsf{CNM}^{(k)}$-attaquant $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ contre π. Nous construisons un $(m,n)-\mathsf{IND}-\mathsf{PA}^{(k)}$-attaquant $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}'_2)$ de la manière suivante :

- $\mathcal{B}_1$ exécute $\mathcal{A}_1$: à chaque fois que $\mathcal{A}_1$ fait une requête de déchiffrement, $\mathcal{B}_1$ fait la même requête à l'oracle de déchiffrement, puis transmet la réponse à $\mathcal{A}_1$. Quand $\mathcal{A}_1$ retourne une distribution $\mathcal{M}$ et l'information s, on choisit aléatoirement deux textes clairs (m_0, m_1) suivant la distribution $\mathcal{M}$. $\mathcal{B}_1$ retourne ces deux textes ainsi que l'information s ;
- Le challengeur choisit aléatoirement $b \xleftarrow{R} \{0,1\}$ et $r \xleftarrow{R} R$, puis fabrique le challenge $c = \mathcal{E}_{\mathsf{pk}}(m_b, r)$;
- $\mathcal{B}_2$ reçoit l'information s et le challenge c. Il les transmet à $\mathcal{A}_2$. $\mathcal{B}_2$ exécute $\mathcal{A}_2$: à chaque fois que $\mathcal{A}_2$ fait une requête de déchiffrement, $\mathcal{B}_2$ fait la même requête à l'oracle de déchiffrement, puis transmet la réponse à $\mathcal{A}_2$. Quand $\mathcal{A}_2$ retourne $(R, \mathbf{y})$, $\mathcal{B}_2$ retourne $(R, \mathbf{y})$. $\mathcal{B}'_2$ reçoit donc le vecteur de textes clairs correspondant $\mathbf{x}$. $\mathcal{B}'_2$ vérifie si $R(\mathbf{x}, m_0)$ est vraie. Si oui, il retourne $b' = 0$, sinon $b' = 1$.

$$\begin{aligned}
\mathsf{Adv}_{\pi}^{(m,n)-\mathsf{ind}-\mathsf{pa}^{(k)}}(\mathcal{B}) &= \Pr[b' = 0 \,|\, b = 0] - \Pr[b' = 0 \,|\, b = b_1] \\
&= \Pr[R(\mathbf{x}, m_0) \,|\, b = 0] - \Pr[R(\mathbf{x}, m_0) \,|\, b = 1] \\
&= \Pr[R(\mathbf{x}, m_0) \,|\, b = 0] - \Pr[R(\mathbf{x}, m_1) \,|\, b = 0] \\
&= \mathsf{Adv}_{\pi}^{(m,n)-\mathsf{ind}-\mathsf{cnm}^{(k)}}(\mathcal{A}).
\end{aligned}$$

Remarquons que le temps de calcul de $\mathcal{B}$ est égal à celui de $\mathcal{A}$, augmenté du temps d'évaluation de R. □

Remarque. Soient les identifications suivantes,

$$(m,n)-\mathsf{IND}-\mathsf{PA}^{(1)} = (m, n+1)-\mathsf{IND} \qquad (m,n)-\mathsf{CNM}^{(1)} = (m,n)-\mathsf{NM},$$

on a $(m, n+1)-\mathsf{IND} = (m,n)-\mathsf{NM}$.

2.2.2 Relations entre les notions de sécurité revisitées

Dans [7], les auteurs ont montré plusieurs relations entre les notions de sécurité. Leur résultats sont résumés dans la figure 2.1 :

On peut voir que tous ces résultats se déduisent de notre théorème 15. En effet, les implications et séparations non triviales peuvent être montrées de la façon suivante :

1. $\mathsf{IND}-\mathsf{CCA1} \not\rightarrow \mathsf{NM}-\mathsf{CPA}$: il s'agit en effet de
$$(N,0)-\mathsf{IND} \not\rightarrow (0,1)-\mathsf{IND} = (0,0)-\mathsf{NM} = \mathsf{NM}-\mathsf{CPA}.$$
2. $\mathsf{NM}-\mathsf{CPA} \not\rightarrow \mathsf{IND}-\mathsf{CCA1}$: il s'agit en effet de
$$\mathsf{NM}-\mathsf{CPA} = (0,0)-\mathsf{NM} = (0,1)-\mathsf{IND} \not\rightarrow (N,0)-\mathsf{IND}.$$
3. $\mathsf{IND}-\mathsf{CCA2} \rightarrow \mathsf{NM}-\mathsf{CCA2}$: il s'agit en effet de
$$(N,M+1)-\mathsf{IND} = (N,M)-\mathsf{NM}.$$
4. $\mathsf{NM}-\mathsf{CCA1} \not\rightarrow \mathsf{NM}-\mathsf{CCA2}$: il s'agit en effet de
$$(N,0)-\mathsf{NM} = (N,1)-\mathsf{IND} \not\rightarrow (0,2)-\mathsf{IND} = (0,1)-\mathsf{NM} \not\rightarrow (N,M)-\mathsf{NM}.$$

2.2.3 Nouveau modèle d'attaque : CCAO2

En guise d'application, nous introduisons un nouveau modèle d'attaque appelé *attaque à chiffrés choisis post-challenge* et dénoté CCAO2 (pour indiquer des attaques à chiffrés choisis mais seulement dans la deuxième étape, *i.e.* après la réception du challenge.) Ce nouveau scénario complète la figure 2.1 avec la notion de sécurité de $(0, \mathrm{poly}(\cdot))-\mathsf{ind}$. Par ailleurs, d'un point de vue pratique, il modélise des situations réalistes puisque le contrôle dont dispose l'attaquant sur la distribution a priori des clairs est limité. De plus, il couvre également les situations où l'attaquant ne commence l'attaque qu'après avoir pris conscience de l'importance d'un chiffré spécifique.

Grâce au théorème 15, nous montrons que ce modèle d'attaque est indépendant d'autres modèles connus dans la littérature.

Définition 17 (Attaques post-challenge) *Un attaquant est appelé* attaquant à chiffrés choisis post-challenge *(dénoté* CCAO2*-attaquant) s'il ne peut accéder à l'oracle de déchiffrement qu'après que le challenge est connu et qu'il est contraint de ne pas faire de requête sur le challenge à cet oracle.*

Ce nouveau modèle d'attaque, combiné avec les objectifs classiques de sécurité, nous donne les deux notions de sécurité : IND−CCAO2 et NM−CCAO2. Elles sont indépendantes des notions précédentes, à l'exception des implications triviales. D'abord, il est clair que, pour n'importe quel XXX, XXX−CCA2 implique XXX−CCA1 et XXX−CCAO2. Cependant, grâce au résultat ci-dessus, nous prouvons que l'inverse n'est pas vrai. En effet, nous avons les corollaires suivants :

Corollaire 18 IND−CCAO2 *et* IND−CCA1 *sont deux notions indépendantes. En d'autres termes, sous l'hypothèse de l'existence de permutations à sens-unique à trappe, il existe un schéma* IND−CCA1 *sûr mais pas* IND−CCAO2 *sûr. De même, il existe un schéma* IND−CCAO2 *sûr mais pas* IND−CCA1 *sûr.*

Corollaire 19 IND−CCA1 *et* IND−CCAO2 *n'impliquent pas, même pris ensemble,* IND−CCA2*. En d'autres termes, sous l'hypothèse de l'existence de permutations à sens-unique à trappe, il existe un schéma à la fois* IND−CCA1 *sûr et* IND−CCAO2 *sûr mais pas* IND−CCA2 *sûr.*

3

Analyse des notions de sécurité pour le chiffrement symétrique

Sommaire

Bellare *et al.* [6] ont étudié plusieurs variantes de la notion d'indistinguabilité pour le chiffrement symétrique telles que *find-then-guess*, *left-or-right* et *real-or-random*, et des relations entre ces variantes. Katz et Yung [72] ont étudié la différence réelle entre ces divers genres d'attaques, contre le chiffrement symétrique probabiliste. Ils ont prouvé qu'une attaque adaptative à clairs choisis (où des requêtes sont autorisées même après que le challenge est connu) n'aide pas plus qu'une attaque non-adaptative à clairs choisis (ou attaque *lunchtime*, où les accès à l'oracle sont limités jusqu'à réception du challenge).

Pour le chiffrement par bloc, qui est un chiffrement déterministe et préserve la longueur, la notion de sécurité sémantique a été récemment étudiée par Desai et Miner [40]. Ils ont affirmé l'équivalence entre la propriété de permutation pseudo-aléatoire et la sécurité sémantique contre des attaques à clairs choisis.

Dans notre travail [97], nous considérons la notion d'indistinguabilité pour le chiffrement par bloc. Remarquons que l'inverse d'un chiffrement par bloc est aussi un chiffrement par bloc, grâce au déterminisme des chiffrements par bloc.

Nous montrons d'abord que la notion d'indistinguabilité, modélisée par le jeu de *find-then-guess* (avec quelques contraintes naturelles) est équivalente à la définition de la sécurité sémantique.

Nous prouvons ensuite que le résultat de Katz et Yung [72] pour le chiffrement probabiliste reste inchangé pour le chiffrement par bloc : une attaque adaptative à clairs choisis n'est pas plus utile qu'une attaque non-adaptative à clairs choisis. De manière plus intéressante, nous considérons aussi les attaques à chiffrés choisis en réduisant des attaques adaptatives à chiffrés choisis à des attaques non-adaptatives à chiffrés choisis. Nous prouvons qu'une attaque adaptative à chiffrés choisis n'aide pas plus qu'une attaque non-adaptative à chiffrés choisis si le chiffrement et son inverse (le déchiffrement) sont sûrs contre des attaques non-adaptatives à chiffrés choisis.

Nous considérons finalement la relation entre la notion d'indistinguabilité proposée ci-dessus et la notion classique, *i.e.* permutation super pseudo-aléatoire. Nous montrons que pour le chiffrement par bloc, la propriété de permutation super pseudo-aléatoire et l'indistinguabilité contre des attaques non-adaptatives (clairs choisis + chiffrés choisis) sont équivalentes, sous l'hypothèse que le chiffrement et le déchiffrement aient le même niveau de sécurité contre les attaques non-adaptatives.

Ces résultats ont des implications importantes en pratique car, pour le chiffrement par bloc, le chiffrement et le déchiffrement sont souvent très similaires pour des raisons d'efficacité. À titre d'exemple, considérons le cas de DES. Sous la conjecture que la rotation à gauche dans la dérivation des sous-clefs (« key schedule » en anglais) peut être remplacée par une rotation à droite, nous prouvons que les résultats ci-dessus sont valables sans aucune hypothèse supplémentaire.

3.1 Notions de sécurité pour le chiffrement symétrique

3.1.1 Schéma de chiffrement symétrique

Pour commencer, on rappelle la définition formelle d'un schéma de chiffrement symétrique $\pi = (k, \ell, \mathcal{E}, \mathcal{D})$. Il est défini par deux algorithmes, paramétrés par une clef k supposée uniformément distribuée dans $\{0,1\}^k$. Remarquons qu'en pratique, deux informations principales sont k et ℓ, le nombre de bits, respectivement, de la clef et du bloc à être chiffré :

- l'algorithme de chiffrement $\mathcal{E}_{\mathsf{k}}$, qui reçoit comme entrée un message m de l'ensemble $\{0,1\}^{\ell}$ et un aléa r de $\{0,1\}^{\mu}$ et retourne un texte chiffré c dans $\{0,1\}^{\nu} : c = \mathcal{E}_{\mathsf{k}}(m,r)$.
- l'algorithme de déchiffrement $\mathcal{D}_{\mathsf{k}}$, qui reçoit comme entrée un texte chiffré c, et retourne le texte clair correspondant $m : m = \mathcal{D}_{\mathsf{k}}(c)$, ou $\perp$ si ce texte clair n'existe pas " C'est typiquement un algorithme déterministe.

La condition de consistance requise est que le déchiffrement d'un chiffrement redonne le texte clair original : pour tout clef k, $\mathcal{D}_{\mathsf{k}}(\mathcal{E}_{\mathsf{k}}(m,r)) = m$ pour tout $m \in \{0,1\}^{\ell}$.

3.1.2 Chiffrement par bloc

Le chiffrement par bloc est un cas particulier du chiffrement symétrique qui est déterministe et préserve la longueur. Le chiffrement n'utilise donc pas d'aléa. De plus, comme il préserve la longueur, tout texte chiffré est valide : il s'agit d'une permutation pour chaque clef (ainsi, $\mu = 0$ et $\nu = \ell$.) Étant donné un chiffrement par bloc $\pi = (k, \ell, \mathcal{E}, \mathcal{D})$, on définit son inverse par :

$$\pi^{-1} = (k, \ell, \mathcal{E}^{-1} = \mathcal{D}, \mathcal{D}^{-1} = \mathcal{E}).$$

On remarque que π^{-1} est aussi un chiffrement par bloc.

3.1.3 Sécurité sémantique

La notion naturelle de sécurité pour le chiffrement par bloc est la variante calculatoire de la sécurité parfaite : la vue d'un texte chiffré ne peut pas servir à apprendre d'information sur le texte clair. Cela est formalisé par la notion de *sécurité sémantique* [59], dans laquelle un SEM-attaquant $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ joue le jeu en deux étapes suivant :

- une clef k est uniformément choisie dans $\{0,1\}^{k}$;
- Étape 1 : $\mathcal{A}_1$ retourne une distribution échantillonnable D sur l'ensemble $\{0,1\}^{\ell}$ et un état d'information s à faire suivre à la deuxième étape.
- un message m est choisi dans $\{0,1\}^{\ell}$ selon la distribution D (dénoté $m \xleftarrow{D} \{0,1\}^{\ell}$), et le challenge est calculé par $c = \mathcal{E}_{\mathsf{k}}(m)$;
- Étape 2 : le challenge c et l'état d'information s sont donnés à $\mathcal{A}_2$. Il retourne un prédicat calculable f.

L'attaquant gagne le jeu si $f(m)$ est vrai, *i.e.* il est capable d'apprendre au moins un bit d'information sur m à partir du texte chiffré c. Cependant, un attaquant peut gagner tout le temps en retournant un prédicat trivial, constant par exemple. Alors, on dit que $\mathcal{A}$ casse la sécurité sémantique si le prédicat f est vrai sur m avec une probabilité significativement plus grande que pour un autre texte clair aléatoire m' (suivant la même distribution « a priori » D). Formellement, on définit l'avantage $\mathsf{Adv}^{\mathsf{sem}}_{\pi}(\mathcal{A})$ d'un attaquant $\mathcal{A}$, contre le

schéma de chiffrement π au sens de la sécurité sémantique par :

$$\Pr\left[\begin{array}{l}\mathsf{k} \xleftarrow{R} \{0,1\}^k; (D,s) \leftarrow \mathcal{A}_1();\\ m, m' \xleftarrow{D} \{0,1\}^\ell;\\ c = \mathcal{E}_\mathsf{k}(m); f \leftarrow \mathcal{A}_2(s,c) :\\ \qquad f(m) = 1\end{array}\right] - \Pr\left[\begin{array}{l}\mathsf{k} \xleftarrow{R} \{0,1\}^k; (D,s) \leftarrow \mathcal{A}_1();\\ m, m' \xleftarrow{D} \{0,1\}^\ell;\\ c = \mathcal{E}_\mathsf{k}(m); f \leftarrow \mathcal{A}_2(s,c) :\\ \qquad f(m') = 1\end{array}\right].$$

Définition 20 *Un schéma de chiffrement π est (ε, t)-sémantiquement sûr si pour tout attaquant $\mathcal{A}$ qui fonctionne en temps t, $\mathsf{Adv}^{\mathsf{sem}}_\pi(\mathcal{A}) \leq \varepsilon$.*

Attaquants

L'attaquant $\mathcal{A}$ peut avoir accès à d'autres informations supplémentaires que le challenge, telles que des couples de texte clair-texte chiffré. Plusieurs genres d'attaques peuvent être mis en place en fonction de la manière selon laquelle ces couples sont définis : attaques à clairs choisis et/ou à chiffrés choisis. De plus, le choix du texte clair ou chiffré peut être autorisé soit uniquement avant la réception du challenge, soit à n'importe quel moment. Une telle information additionnelle est modélisée par un accès (il)limité aux oracles qui calculent des chiffrements ou des déchiffrements. Un (t, e_1, d_1, e_2, d_2)-attaquant $\mathcal{A} = (\mathcal{A}_1^{\mathcal{E}_\mathsf{k},\mathcal{D}_\mathsf{k}}, \mathcal{A}_2^{\mathcal{E}_\mathsf{k},\mathcal{D}_\mathsf{k}})$ est un attaquant qui mène une attaque en deux étapes $\mathcal{A}$, où $\mathcal{A}_1$ ($\mathcal{A}_2$ resp.) peut faire au maximum e_1 et d_1 (e_2 et d_2 resp.) requêtes aux oracles de chiffrement $\mathcal{E}_\mathsf{k}$ et de déchiffrement $\mathcal{D}_\mathsf{k}$. De cette manière, on considère deux cas : un attaquant passif, où $e_1 = e_2 = d_1 = d_2 = 0$ (dénoté $\mathsf{P0{-}C0}$), ou un attaquant actif (dénoté $\mathsf{P}X{-}\mathsf{C}Y$), selon l'accès aux oracles :

$X =$ '1' – $e_1 > 0$ mais $e_2 = 0$,	clairs choisis non-adaptatifs ($\mathsf{P1{-}C}Y$) ;
$Y =$ '1' – $d_1 > 0$ mais $d_2 = 0$,	chiffrés choisis non-adaptatifs ($\mathsf{P}X{-}\mathsf{C1}$) ;
$X =$ '2' – $e_2 > 0$ quel que soit e_1,	clairs choisis adaptatifs ($\mathsf{P2{-}C}Y$) ;
$Y =$ '2' – $d_2 > 0$ quel que soit d_1,	chiffrés choisis adaptatifs ($\mathsf{P}X{-}\mathsf{C2}$).

On rappelle que tous les attaquants sont adaptatifs dans la mesure où une nouvelle requête peut dépendre des réponses précédemment données par les oracles. Ainsi, par « adaptatif » on comprend « challenge-adaptatif », alors que « non-adaptatif » signifie « challenge non-adaptatif ». Un tel $\mathsf{P}X{-}\mathsf{C}Y$ attaquant peut jouer le jeu contre la sécurité sémantique, mais il y a des contraintes naturelles dans le cas d'accès aux oracles. Ces contraintes sont les suivantes :

- si l'attaquant a accès à l'oracle de déchiffrement (soit $\mathsf{C1}$ ou $\mathsf{C2}$), il est contraint de ne pas demander le challenge c dans la deuxième étape ;
- si l'attaquant a accès à l'oracle de chiffrement (soit $\mathsf{P1}$ ou $\mathsf{P2}$), le support S_D de D (l'ensemble des textes clairs qui ont une probabilité non nulle dans D) doit être disjoint de la liste des textes clairs demandés à l'oracle de chiffrement, ou obtenus par l'oracle de déchiffrement au cours de la première étape.

La première contrainte est classique et la deuxième est naturelle dans le cas considéré du chiffrement déterministe. Nous montrerons (en prouvant l'équivalence entre cette notion et celle de *find-then-guess*) que cette deuxième contrainte est une contrainte minimale.

Définition 21 *Un schéma de chiffrement π est dit $(\varepsilon, t, e_1, d_1, e_2, d_2)$-sémantiquement sûr si pour tout $(t, e_1, d_1, e_2, d_2)-\mathsf{SEM}$ attaquant $\mathcal{A}$, qui fait au plus e_1 et d_1 (e_2 et d_2 resp.) requêtes de chiffrement et de déchiffrement dans la première étape (resp. dans la deuxième étape) en temps t, $\mathsf{Adv}_{\pi}^{\mathsf{sem}}(\mathcal{A}) \leq \varepsilon$.*

3.1.4 Indistinguabilité : « find-then-guess »

La notion d'*indistinguabilité* (aussi connue sous le nom de *find-then-guess* [6]) étudie le cas d'un $(t, e_1, d_1, e_2, d_2)-\mathsf{IND}$ attaquant $\mathcal{A} = (\mathcal{A}_1^{\mathcal{E}_\mathsf{k}, \mathcal{D}_\mathsf{k}}, \mathcal{A}_2^{\mathcal{E}_\mathsf{k}, \mathcal{D}_\mathsf{k}})$ qui joue le jeu suivant :

- une clef k est uniformément choisie dans $\{0,1\}^k$;
- étape 1 (choix) : $\mathcal{A}_1^{\mathcal{E}_\mathsf{k}, \mathcal{D}_\mathsf{k}}$ retourne deux textes clairs (m_0, m_1), et un état d'information s à faire suivre à la deuxième étape d'attaque ;
- un bit b est aléatoirement choisi et le challenge est calculé par $c = \mathcal{E}_\mathsf{k}(m_b)$;
- étape 2 (estimation) : L'état d'information s et le texte chiffré c sont donnés à $\mathcal{A}_2^{\mathcal{E}_\mathsf{k}, \mathcal{D}_\mathsf{k}}$. Il retourne son estimation b' pour b.

L'attaquant gagne le jeu si $b' = b$, ce qui veut dire qu'il est capable de distinguer le chiffrement de m_0 de celui de m_1. Cependant, tout attaquant peut gagner le jeu une fois sur deux en jouant simplement à pile ou face. C'est pour cette raison qu'on dit que $\mathcal{A}$ casse la sécurité *find-then-guess* si $b' = b$ avec une probabilité significativement plus grande que $1/2$. Formellement, l'avantage dont dispose un attaquant $\mathcal{A}$ face à la sécurité *find-then-guess*, ou l'*indistinguabilité* d'un schéma de chiffrement π, est défini par :

$$\mathsf{Adv}_{\pi}^{\mathsf{ind}}(\mathcal{A}) = 2 \times \Pr\left[\begin{array}{l} \mathsf{k} \xleftarrow{R} \{0,1\}^k; (m_0, m_1, s) \leftarrow \mathcal{A}_1^{\mathcal{E}_\mathsf{k}, \mathcal{D}_\mathsf{k}}(); b \xleftarrow{R} \{0,1\}; \\ c = \mathcal{E}_\mathsf{k}(m_b); b' \leftarrow \mathcal{A}_2^{\mathcal{E}_\mathsf{k}, \mathcal{D}_\mathsf{k}}(s, c) : b' = b \end{array}\right] - 1.$$

Comme dans la définition de la sécurité sémantique, il y a aussi des contraintes naturelles en cas d'accès aux oracles :

- si l'attaquant a l'accès à l'oracle de déchiffrement ($\mathsf{C1}$ ou $\mathsf{C2}$), il est contraint de ne pas demander le challenge c dans la deuxième étape ;
- si l'attaquant a l'accès à l'oracle de chiffrement ($\mathsf{P1}$ ou $\mathsf{P2}$), il est contraint de ne pas soumettre ni m_0 ni m_1 à l'oracle de chiffrement à n'importe quel moment, et de ne pas obtenir m_0 or m_1 de l'oracle de déchiffrement au cours de la première étape.

Ces contraintes peuvent être ré-écrites :

$$m_0, m_1 \notin \Lambda_{\mathcal{E}}^m \qquad c \notin \Lambda_{\mathcal{D}}^c.$$

où $\Lambda_{\mathcal{E}}$ ($\Lambda_{\mathcal{D}}$ resp.) sont les listes de couples de texte clair-texte chiffré (m, c) obtenus par l'oracle de chiffrement (et l'oracle de déchiffrement resp.). Les exposants supérieurs m et c sont utilisés pour restreindre ces listes à la première coordonnée (la deuxième resp.). On obtient donc deux listes de texte clairs $\Lambda_{\mathcal{E}}^m$ et $\Lambda_{\mathcal{D}}^m$, et deux listes de texte chiffrés $\Lambda_{\mathcal{E}}^c$ et $\Lambda_{\mathcal{D}}^c$.

Définition 22 *Un schéma de chiffrement π est $(\varepsilon, t, e_1, d_1, e_2, d_2)$-indistinguable si pour tout $(t, e_1, d_1, e_2, d_2)-\mathsf{IND}$-attaquant $\mathcal{A}$ qui peut faire au plus e_1 et d_1 (resp. e_2 et d_2) requêtes de chiffrement et de déchiffrement à la première étape (resp. à la deuxième étape) en temps t, $\mathsf{Adv}_{\pi}^{\mathsf{ind}}(\mathcal{A}) \leq \varepsilon$.*

3.1.5 Permutations pseudo-aléatoires et super pseudo-aléatoires

On appelle $\mathcal{SP}_\ell$ l'ensemble de toutes les permutations sur $\{0,1\}^\ell$. On appelle une permutation véritablement aléatoire une permutation aléatoirement choisi dans l'ensemble $\mathcal{SP}_\ell$.

Permutations pseudo-aléatoires

Les permutations pseudo-aléatoires sont un cas particulier des fonctions pseudo-aléatoires où les fonctions sont des permutations. Cette notion est formalisée comme suit : si un attaquant $\mathcal{A}$ a accès à un oracle $\mathcal{O}_b$ ($\mathcal{O}_0$ correspond à une permutation véritablement aléatoire $\mathcal{P}$ et $\mathcal{O}_1$ correspond au chiffrement $\mathcal{E}_\mathsf{k}$ avec une clef aléatoire k), il ne peut pas deviner b, *i.e.* son avantage défini ci-dessous doit être négligeable :

$$\mathsf{Adv}_{\pi}^{\mathsf{prp}}(\mathcal{A}) = 2 \times \Pr\left[\begin{array}{l} \mathsf{k} \xleftarrow{R} \{0,1\}^k; \mathcal{P} \xleftarrow{R} \mathcal{SP}_\ell; \mathcal{O}_0 = \mathcal{P}; \mathcal{O}_1 = \mathcal{E}_\mathsf{k}; \\ b \xleftarrow{R} \{0,1\}; b' \leftarrow \mathcal{A}^{\mathcal{O}_b}() : b' = b \end{array}\right] - 1.$$

Définition 23 *Un schéma de chiffrement π est dit (ε, t, n)-permutation pseudo-aléatoire, dénoté $(\varepsilon, t, n)-\mathsf{PRP}$ si pour tout $(t, n)-\mathsf{PRP}$ attaquant $\mathcal{A}$ qui demande au plus n requêtes de chiffrement en temps t, $\mathsf{Adv}_{\pi}^{\mathsf{prp}}(\mathcal{A}) \leq \varepsilon$.*

Permutations super pseudo-aléatoires

La notion de permutations pseudo-aléatoires ne fournit pas l'accès à l'oracle de déchiffrement. Cette restriction est levée dans une notion plus forte : la notion de permutations super pseudo-aléatoires. Ainsi, l'attaquant, quand il a accès à un oracle, ne sait pas s'il s'agit d'une permutation parfaitement aléatoire $\mathcal{P}$ ou d'un algorithme de chiffrement $\mathcal{E}_\mathsf{k}$ avec une clef aléatoire. Cependant, dans le cadre de permutations super pseudo-aléatoires, l'attaquant a accès non seulement à la permutation $\mathcal{O}_b$, qui est soit $\mathcal{P}$ soit $\mathcal{E}_\mathsf{k}$, mais également à son inverse $\mathcal{O}_b^{-1}$, qui est soit $\mathcal{P}^{-1}$ soit $\mathcal{D}_\mathsf{k}$. On mesure la qualité d'un attaquant par :

$$\mathsf{Adv}_{\pi}^{\mathsf{sprp}}(\mathcal{A}) = 2 \times \Pr\left[\begin{array}{l} \mathsf{k} \xleftarrow{R} \{0,1\}^k; \mathcal{P} \xleftarrow{R} \mathcal{SP}_\ell; \\ (\mathcal{O}_0, \mathcal{O}_0^{-1}) = (\mathcal{P}, \mathcal{P}^{-1}); (\mathcal{O}_1, \mathcal{O}_1^{-1}) = (\mathcal{E}_\mathsf{k}, \mathcal{D}_\mathsf{k}); \\ b \xleftarrow{R} \{0,1\}; b' \leftarrow \mathcal{A}^{\mathcal{O}_b, \mathcal{O}_b^{-1}}() : b' = b \end{array}\right] - 1.$$

Définition 24 *Un schéma de chiffrement π est dit (ε, t, n, m)-permutation super pseudo-aléatoire, dénoté $(\varepsilon, t, n, m)-\mathsf{SPRP}$ si pour tout $(t, n, m)-\mathsf{SPRP}$ attaquant $\mathcal{A}$ qui fait au plus n requêtes de chiffrement et au plus m requêtes de déchiffrement en temps t, $\mathsf{Adv}^{\mathsf{sprp}}_{\pi}(\mathcal{A}) \leq \varepsilon$.*

3.1.6 Equivalences

On rappelle que l'indistinguabilité et la sécurité sémantique sont des notions équivalentes si D est efficacement échantillonnable et que le prédicat f est efficacement calculable. Dans le cas particulier du chiffrement par bloc, nous précisons le théorème suivant :

Théorème 25 *Pour tout schéma de chiffrement $\pi = (k, \ell, \mathcal{E}, \mathcal{D})$:*

$$\frac{1}{2} \times \mathsf{Adv}^{\mathsf{ind}}_{\pi}(t, e_1, d_1, e_2, d_2) \leq \mathsf{Adv}^{\mathsf{sem}}_{\pi}(t, e_1, d_1, e_2, d_2) \leq \mathsf{Adv}^{\mathsf{ind}}_{\pi}(t', e_1, d_1, e_2, d_2),$$

où $t' \leq t + 2T_D + T_f$, si le temps d'échantillonnage pour D est borné par T_D et le temps pour évaluer le prédicat f est borné par T_f.

Preuve.

Considérons $\mathcal{A} = (\mathcal{A}_1^{\mathcal{E}_{\mathsf{k}}, \mathcal{D}_{\mathsf{k}}}, \mathcal{A}_2^{\mathcal{E}_{\mathsf{k}}, \mathcal{D}_{\mathsf{k}}})$ un $(t, e_1, d_1, e_2, d_2)-\mathsf{IND}$-attaquant contre le schéma π. Nous construisons $\mathcal{B} = (\mathcal{B}_1^{\mathcal{E}_{\mathsf{k}}, \mathcal{D}_{\mathsf{k}}}, \mathcal{B}_2^{\mathcal{E}_{\mathsf{k}}, \mathcal{D}_{\mathsf{k}}})$, un $(t, e_1, d_1, e_2, d_2)-\mathsf{SEM}$-attaquant, contre π de la manière suivante :

- $\mathsf{k} \xleftarrow{R} \{0,1\}^k$;
- $\mathcal{B}_1$ exécute $\mathcal{A}_1$: à chaque fois que $\mathcal{A}_1$ fait une requête de chiffrement ou de déchiffrement, $\mathcal{B}_1$ fait la même requête à l'oracle de chiffrement ou de déchiffrement, puis transmet la réponse à $\mathcal{A}_1$. Quand $\mathcal{A}_1$ retourne deux candidats (m_0, m_1) et s, $\mathcal{B}_1$ définit et retourne une distribution uniforme $\mathcal{D} = \{m_0, m_1\}$ et l'information s ;
- $m \xleftarrow{D} \{0,1\}^{\ell}$, et $c = \mathcal{E}_{\mathsf{k}}(m; r)$ (*i.e.* choisir $b \xleftarrow{R} \{0,1\}$, poser $m = m_b$ puis calculer $c = \mathcal{E}_{\mathsf{k}}(m_b)$) ;
- $\mathcal{B}_2$ reçoit l'information s et le challenge c. Il les transmet à $\mathcal{A}_2$. $\mathcal{B}_2$ exécute $\mathcal{A}_2$: à chaque fois que $\mathcal{A}_2$ fait une requête de chiffrement ou de déchiffrement, $\mathcal{B}_2$ fait la même requête à l'oracle de chiffrement ou de déchiffrement, puis transmet la réponse à $\mathcal{A}_2$. Quand $\mathcal{A}_2$ devine b' pour b, $\mathcal{B}_2$ définit $f(x)$ le prédicat $x = m_{b'}$.

Par définition, si on choisit m' selon la distribution $\mathcal{D}$, ce qui est équivalent à choisir aléatoirement un bit d indépendant de b et b', et puis poser $m' = m_d$, alors :

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{A}) &= 2 \times \Pr[b' = b] - 1 \\
\mathsf{Adv}^{\mathsf{sem}}_{\pi}(\mathcal{B}) &= \Pr[f(m)] - \Pr[f(m')] = \Pr[m_b = m_{b'}] - \Pr[m_d = m_{b'}] \\
&= \Pr[b = b'] + \Pr[b \neq b' \wedge m_0 = m_1] - \Pr[d = b'] - \Pr[d \neq b' \wedge m_0 = m_1] \\
&= \left(\Pr[b = b'] - \frac{1}{2}\right) + \left(\Pr[b \neq b' \mid m_0 = m_1] - \frac{1}{2}\right) \times \Pr[m_0 = m_1].
\end{aligned}$$

Remarquons que si $m_0 = m_1$ alors, du point de vue de la théorie de l'information, l'attaquant n'a aucune information sur b, b' est donc parfaitement indépendant de b : $\mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{A}) = 2\times\mathsf{Adv}^{\mathsf{sem}}_{\pi}(\mathcal{B})$. De plus, le temps de calcul de $\mathcal{B}$ est exactement le même que celui de $\mathcal{A}$ car aucun calcul additionnel n'est exigé, alors : $\mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{A}) \leq 2\times\mathsf{Adv}^{\mathsf{sem}}_{\pi}(t, e_1, d_1, e_2, d_2)$.

Concernant les restrictions, si $\mathcal{A}$ a accès à l'oracle de déchiffrement, il ne peut faire de requête sur le challenge c. Par conséquent, $\mathcal{B}$ n'en fait pas non plus. De plus, $\mathcal{A}$ est aussi interdit de requêtes sur m_0 ou m_1 à l'oracle de chiffrement. Alors, $\mathcal{B}$ n'en fait pas non plus car ces deux messages sont les seuls à probabilité non nulle dans la distribution $\mathcal{D}$. En bref, $\mathcal{B}$ ne viole aucune restriction.

On passe maintenant à la relation de droite du théorème.

Considérons un $(t, e_1, d_1, e_2, d_2)-\mathsf{SEM}$-attaquant $\mathcal{A} = (\mathcal{A}_1^{\mathcal{E}_\mathsf{k},\mathcal{D}_\mathsf{k}}, \mathcal{A}_2^{\mathcal{E}_\mathsf{k},\mathcal{D}_\mathsf{k}})$ contre π. Nous construisons un $(t', e_1, d_1, e_2, d_2)-\mathsf{IND}$-attaquant $\mathcal{B} = (\mathcal{B}_1^{\mathcal{E}_\mathsf{k},\mathcal{D}_\mathsf{k}}, \mathcal{B}_2^{\mathcal{E}_\mathsf{k},\mathcal{D}_\mathsf{k}})$ de la manière suivante :

- $\mathsf{k} \xleftarrow{R} \{0,1\}^k$;
- $\mathcal{B}_1$ exécute $\mathcal{A}_1$: à chaque fois que $\mathcal{A}_1$ fait une requête de chiffrement ou de déchiffrement, $\mathcal{B}_1$ fait la même requête à l'oracle de chiffrement ou de déchiffrement, puis transmet la réponse à $\mathcal{A}_1$. Quand $\mathcal{A}_1$ retourne une distribution $\mathcal{D}$ et l'information s, on choisit aléatoirement deux textes clairs (m_0, m_1) selon $\mathcal{D}$. $\mathcal{B}_1$ les retourne avec l'information s ;
- $b \xleftarrow{R} \{0,1\}$ et $c = \mathcal{E}_\mathsf{k}(m_b)$
- $\mathcal{B}_2$ reçoit l'information s et le challenge c. Il les transmet à $\mathcal{A}_2$. $\mathcal{B}_2$ exécute $\mathcal{A}_2$: à chaque fois que $\mathcal{A}_2$ fait une requête de chiffrement ou de déchiffrement, $\mathcal{B}_2$ fait la même requête à l'oracle de chiffrement ou de déchiffrement et puis transmet la réponse à $\mathcal{A}_2$. Quand $\mathcal{A}_2$ retourne f, $\mathcal{B}_2$ vérifie si $f(m_0)$ est vrai. Si oui, il retourne $b' = 0$, sinon il retourne $b' = 1$.

On choisit m' selon la distribution $\mathcal{D}$ tel que $m' \neq m_b$, ce qui est équivalent à poser $m' = m_{\overline{b}}$ (où $\overline{b} = 1 - b$). On dénote E, l'événement où $f(m_0) = f(m_1)$. Par définition :

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{sem}}_{\pi}(\mathcal{A}) &= \Pr[f(m_b)] - \Pr[f(m')] = \Pr[f(m_b)] - \Pr[f(m_{\overline{b}})] \\
&= \Pr[f(m_b) \wedge \neg\mathsf{E}] - \Pr[f(m_{\overline{b}}) \wedge \neg\mathsf{E}] \\
&= \Pr[\neg\mathsf{E}] \times (\Pr[f(m_b) \mid \neg\mathsf{E}] - \Pr[\neg f(m_b) \mid \neg\mathsf{E}]) \\
\mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{B}) &= 2 \times \Pr[b' = b] - 1 = \Pr[f(m_0) \mid b = 0] - \Pr[f(m_0) \mid b = 1] \\
&= \Pr[f(m_b) \wedge \mathsf{E} \mid b = 0] + \Pr[f(m_b) \wedge \neg\mathsf{E} \mid b = 0] \\
&\quad - \Pr[f(m_b) \wedge \mathsf{E} \mid b = 1] - \Pr[\neg f(m_b) \wedge \neg\mathsf{E} \mid b = 1].
\end{aligned}$$

La sortie de $\mathcal{A}$ ne dépend pas de la valeur b, mais seulement de c, alors :

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{B}) &= \Pr[f(m_b) \wedge \neg\mathsf{E}] - \Pr[\neg f(m_b) \wedge \neg\mathsf{E}] \\
&= \Pr[\neg\mathsf{E}] \times (\Pr[f(m_b) \mid \neg\mathsf{E}] - \Pr[\neg f(m_b) \mid \neg\mathsf{E}]) = \mathsf{Adv}^{\mathsf{sem}}_{\pi}(\mathcal{A}).
\end{aligned}$$

Remarquons que le temps de calcul de $\mathcal{B}$ est égal à celui de $\mathcal{A}$, plus de deux échantillonnages de $\mathcal{D}$ et d'une évaluation de f. Alors, si le temps de calcul de $\mathcal{B}$ est borné par t, un

échantillonnage de $\mathcal{D}$ est borné par T_D et une évaluation de f est bornée par T_f, on a :

$$\mathsf{Adv}^{\mathsf{sem}}_{\pi}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{ind}}_{\pi}(t + 2T_D + T_f, e_1, d_1, e_2, d_2).$$

Concernant les restrictions, si $\mathcal{A}$ a accès à l'oracle de déchiffrement, il ne peut faire de requête sur le challenge c. Par conséquent, $\mathcal{B}$ n'en fait pas non plus. De plus, $\mathcal{A}$ est aussi interdit de requêtes sur S_D, dont m_0 et m_1, à l'oracle de chiffrement. Alors, $\mathcal{B}$ ne fait pas non plus des requêtes à l'oracle de chiffrement sur m_0 et m_1 En bref, $\mathcal{B}$ ne viole aucune restriction. □

3.2 Résultats sur l'indistinguabilité des chiffrements par bloc

Comme on l'a déjà remarqué, contrairement au cas probabiliste, les contraintes concernent non seulement le challenge mais aussi les messages m_0 et m_1 : le challenge ne peut être demandé à l'oracle de déchiffrement et les messages m_0 et m_1 ne peuvent être demandés à l'oracle de chiffrement. Par conséquent, $m_0, m_1 \notin \Lambda^m_{\mathcal{E}}$ et $c \notin \Lambda^c_{\mathcal{D}}$.

3.2.1 Attaquant « normal »

Dans la suite, tout attaquant est supposé se comporter comme un attaquant *normal*, *i.e.* :

- chaque requête est soumise au plus une fois ;
- si m a été soumis comme requête de chiffrement (ou à l'oracle $\mathcal{O}_b$) et a obtenu la réponse c, alors une requête sur c ne sera jamais demandée à l'oracle de déchiffrement (ou à l'oracle $\mathcal{O}_b^{-1}$) ;
- si c a été demandé comme requête de déchiffrement (ou à l'oracle $\mathcal{O}_b^{-1}$) et a obtenu la réponse m, alors une requête sur m ne sera jamais demandée à l'oracle de chiffrement (ou à l'oracle $\mathcal{O}_b$) ;
- un $(t, n)-\mathsf{PRP}$ attaquant (ou $(t, n, m)-\mathsf{SPRP}$ attaquant, resp.) demande exactement n requêtes à l'oracle $\mathcal{O}_b$ (resp. n requêtes à l'oracle $\mathcal{O}_b$ et m requêtes à l'oracle $\mathcal{O}_b^{-1}$).

Proposition 26 *Tout attaquant peut être rendu* normal *(simplement grâce aux consultations additionnelles dans des tables spécifiques.)*

Preuve. Nous montrons une transformation d'un attaquant $\mathcal{A}$ en un attaquant normal $\mathcal{B}$ sans affaiblir sa puissance. Ceci peut être simplement réalisé grâce à la mémorisation de toutes les requêtes à l'oracle de chiffrement et à l'oracle de déchiffrement, et des réponses correspondantes. En effet, on stocke tous les couples requêtes-réponses (m, c) dans une liste Λ, qui est initialement vide. Quand l'attaquant $\mathcal{A}$ fait une requête de chiffrement

m, $\mathcal{B}$ consulte d'abord dans la liste Λ. Si un élément (m, c) existe, $\mathcal{B}$ retourne immédiatement c à $\mathcal{A}$. Sinon, $\mathcal{B}$ soumet m à l'oracle de chiffrement et retourne à $\mathcal{A}$ la réponse c obtenue de l'oracle de chiffrement et de plus, il ajoute le couple (m, c) dans la liste Λ. $\mathcal{B}$ simule de la même manière pour une requête de déchiffrement de $\mathcal{A}$. Puisque l'algorithme de chiffrement et de déchiffrement sont déterministes, il y a une réponse unique pour chaque requête. La simulation est donc parfaite et le coût de la transformation se résume à quelques consultations dans une liste. Nous montrons aussi la transformation d'un attaquant $\mathcal{A}$ en un attaquant normal $\mathcal{B}$ sans affaiblir sa puissance. □

3.2.2 Attaquant adaptatif

On considère des attaquants dans le cas général dont les accès possibles aux oracles sont caractérisés par des valeurs e_1, d_1, e_2 et d_2. Pour des raisons de clarté, on peut omettre la notation d'oracle dans $\mathcal{A} = (\mathcal{A}_1^{\mathcal{E}_k,\mathcal{D}_k}, \mathcal{A}_2^{\mathcal{E}_k,\mathcal{D}_k})$ et la remplace par $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

Attaques à clairs choisis adaptatives

Tout d'abord, nous revisitons la propriété démontrée par Katz et Yung [72] des schémas de chiffrement symétrique probabiliste. Dans le corollaire 29 ci-dessous, nous prouvons que leur résultat reste vrai pour les chiffrements par bloc. En effet, une fois le challenge connu, un accès à l'oracle de chiffrement n'augmente pas la puissance de l'attaquant qui a déjà accès à cet oracle lors de la première étape.

Théorème 27 *Pour tout chiffrement par bloc π :*

$$\mathsf{Adv}_\pi^{\mathsf{ind}}(t, e_1, d_1, e_2, d_2) \leq (2e_2 + 1) \times \mathsf{Adv}_\pi^{\mathsf{ind}}(t, e_1 + e_2, d_1 + d_2, 0, d_2).$$

Preuve. Considérons $\mathcal{A}$, un $(t, e_1, e_2, d_1, d_2)-\mathsf{IND}$-attaquant, contre le schéma π. D'après la proposition 26, on peut supposer que $\mathcal{A}$ est un attaquant normal. On dénote $\mathcal{A}[\eta_2]$ un nouvel attaquant construit à partir de $\mathcal{A}$, et en restreignant ses interactions : toutes les requêtes faites par $\mathcal{A}_1$ sont transmises ; cependant, dans la deuxième étape, seules les η_2 premières requêtes de chiffrement sont transmises, les autres requêtes de chiffrement sont traitées en retournant des aléas différents des précédents textes chiffrés (les requêtes de déchiffrement et les textes chiffrés retournés comme réponses aux requêtes de chiffrement). On voit facilement que $\mathcal{A}[\eta_2]$ est normal. Remarquons aussi que $\mathcal{A}[e_2] = \mathcal{A}$ et $\mathcal{A}[0]$ est un attaquant qui ne fait aucune requête de chiffrement dans la deuxième étape car toutes les réponses aux requêtes de chiffrement sont des aléas.

Lemme 28 *Pour tout $1 \leq \eta_2 \leq e_2$:*

$$\mathsf{Adv}_\pi^{\mathsf{ind}}(\mathcal{A}[\eta_2]) - \mathsf{Adv}_\pi^{\mathsf{ind}}(\mathcal{A}[\eta_2 - 1]) \leq 2 \times \mathsf{Adv}_\pi^{\mathsf{ind}}(t, e_1 + e_2, d_1 + d_2, 0, d_2). \tag{3.1}$$

Preuve. La preuve complète est présentée en détail dans [97]. L'idée est de construire un $(t, e_1 + \eta_2, d_1 + d_2, 0, d_2)-\mathsf{IND}$ attaquant, dénoté $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$, contre π tel que son avantage est exactement la partie de gauche de la relation 3.1. $\mathcal{B}_1$ exécute exactement $\mathcal{A}_1$, puis simule parfaitement le challengeur : quand $\mathcal{A}_1$ retourne deux candidats (m_0, m_1), il choisit aléatoirement un bit b et retourne le challenge $c = \mathcal{E}_\mathsf{k}(m_b)$. $\mathcal{B}_1$ continue à exécuter $\mathcal{A}_2$ jusqu'à la ${\eta_2}^\mathrm{e}$ requête de chiffrement en utilisant les oracles de chiffrement et de déchiffrement. Au moment où $\mathcal{A}_2$ fait la ${\eta_2}^\mathrm{e}$ requête de chiffrement, $\mathcal{B}_1$ retourne deux candidats q_0, q_1 où q_0 est aléatoirement choisi et q_1 est cette ${\eta_2}^\mathrm{e}$ requête.

$\mathcal{B}_2$ reçoit donc un challenge $a = \mathcal{E}_\mathsf{k}(m_d)$ pour un bit aléatoire d qu'il doit deviner. $\mathcal{B}_2$ exécute $\mathcal{A}_2$ de la façon suivante : d'abord, il utilise son challenge a pour répondre à la ${\eta_2}^\mathrm{e}$ requête de chiffrement ; ensuite, il simule parfaitement les requêtes de déchiffrement en utilisant l'oracle de déchiffrement alors qu'il retourne des aléas pour les requêtes de chiffrement.

Quand $\mathcal{A}_2$ devine b' pour b, $\mathcal{B}_2$ vérifie si $b' = b$: il devine $d' = 1$ si oui, et $d' = 0$ sinon. On remarque que si a est le chiffrement de q_0, l'attaquant $\mathcal{B}$ est $\mathcal{A}[\eta_2 - 1]$ et si a est le chiffrement de q_1, l'attaquant $\mathcal{B}$ est $\mathcal{A}[\eta_2]$. On peut donc montrer que l'avantage pour deviner le bit d de $\mathcal{B}$ est la différence entre l'avantage pour deviner le bit b de $\mathcal{A}[\eta_2]$ et l'avantage de deviner le bit b de $\mathcal{A}[\eta_2 - 1]$. □

En appliquant e_2 fois ce lemme, on obtient facilement :

$$\mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{ind}}_{\pi}(t, e_1, d_1, 0, d_2) + 2e_2 \times \mathsf{Adv}^{\mathsf{ind}}_{\pi}(t, e_1 + e_2, d_1 + d_2, 0, d_2),$$

□

Dans le cas particulier où $d_2 = 0$ (attaques à chiffrés choisis non-adaptatives), on obtient le corollaire suivant : une attaque à clairs choisis adaptative ne donne pas de puissance additionnelle à l'attaquant.

Corollaire 29 *Pour tout chiffrement par bloc π :*

$$\mathsf{Adv}^{\mathsf{ind}}_{\pi}(t, e_1, d_1, e_2, 0) \leq (2e_2 + 1) \times \mathsf{Adv}^{\mathsf{ind}}_{\pi}(t, e_1 + e_2, d_1, 0, 0).$$

Attaques à clairs choisis et à chiffrés choisis adaptatives

Dans cette partie, nous proposons une amélioration du résultat précédent : sous quelques hypothèses spécifiques, un accès adaptatif aux oracles de chiffrement et de déchiffrement n'augmente pas la puissance d'un attaquant qui a déjà accès à ces oracles lors de la première étape. Plus intéressant, le coût de la réduction est linéaire en nombre total de requêtes.

Théorème 30 *Pour tout chiffrement par bloc π : $\mathsf{Adv}^{\mathsf{ind}}_{\pi}(t, e_1, d_1, e_2, d_2)$ a une borne supérieure :*

$$\Big(2(e_2 + d_2) + 1\Big)\left(\begin{array}{c}\mathsf{Adv}^{\mathsf{ind}}_{\pi}(t, e_1 + e_2, d_1 + d_2, 0, 0)\\ +\mathsf{Adv}^{\mathsf{ind}}_{\pi^{-1}}(t, d_1 + d_2 - 1, e_1 + e_2 + 2, 0, 0)\end{array}\right).$$

Preuve. Considérons $\mathcal{A}$, un $(t, e_1, e_2, d_1, d_2)-\mathsf{IND}$-attaquant, contre le schéma π. D'après la proposition 26, on peut supposer que $\mathcal{A}$ est un attaquant normal. Comme dans le théorème 27, on dénote $\mathcal{A}[\eta]$ un nouvel attaquant que l'on construit à partir de $\mathcal{A}$ en restreignant ses interactions : toutes les requêtes sont transmises ; cependant, dans la deuxième étape, seules les η premières requêtes (de chiffrement et de déchiffrement) sont transmises, les autres ont pour réponse des aléas différents des textes précédents dans la même catégorie. S'il s'agit d'une requête de chiffrement, la réponse doit être différente de tous les textes chiffrés précédents (les requêtes de déchiffrement, les chiffres retournés comme réponses aux requêtes de chiffrement et le challenge) " S'il s'agit d'une requête de déchiffrement, la réponse doit être différente de tous les textes clairs précédents (les requêtes de chiffrement, les messages retournés comme réponses aux requêtes de déchiffrement et les deux candidats retournés par $\mathcal{A}_1$). On voit facilement que $\mathcal{A}[\eta]$ est normal. Remarquons aussi que $\mathcal{A}[e_2 + d_2] = \mathcal{A}$ et $\mathcal{A}[0]$ est en effet un attaquant qui ne fait aucune requête de chiffrement dans la deuxième étape car toutes les requêtes de chiffrement ont pour réponse des aléas.

Lemme 31 *Pour tout $\eta \leq e_2 + d_2$: la différence de $\mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{A}[\eta]) - \mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{A}[\eta - 1])$ est bornée par*

$$2 \times \left(\mathsf{Adv}^{\mathsf{ind}}_{\pi}(t, e_1 + e_2, d_1 + d_2, 0, 0) + \mathsf{Adv}^{\mathsf{ind}}_{\pi^{-1}}(t, d_1 + d_2 - 1, e_1 + e_2 + 2, 0, 0) \right),$$

où t est le temps de calcul de $\mathcal{A}$.

Preuve. La preuve complète est présentée en détail dans [97]. L'idée est assez similaire à celle de la preuve du lemme 28 mais à la différence que l'on ne sait pas si la η^{e} requête de $\mathcal{A}_2$ est une requête de chiffrement ou une requête de déchiffrement.

Afin de surmonter ce problème, on construit deux attaquants : $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$, un $(t, e_1 + e_2, d_1 + d_2, 0, 0)-\mathsf{IND}$ attaquant, contre π et $\mathcal{C} = (\mathcal{C}_1, \mathcal{C}_2)$, un $(t, d_1 + d_2 - 1, e_1 + e_2 + 2, 0, 0)-\mathsf{IND}$ attaquant, contre π^{-1}. Les attaquants sont construits de telle manière que pour chaque exécution gagnante de $\mathcal{A}$, celle de $\mathcal{B}$ ou celle de $\mathcal{C}$ est gagnante. Plus précisément, si la η^{e} requête de $\mathcal{A}_2$ est une requête de chiffrement, l'exécution de $\mathcal{B}$ est gagnante et si la η^{e} requête de $\mathcal{A}_2$ est une requête de déchiffrement, l'exécution de $\mathcal{C}$ est gagnante. □

A partir de $\mathcal{A} = \mathcal{A}[e_2 + d_2]$, en appliquant $e_2 + d_2$ fois ce lemme, on obtient :

$$\mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{A}) \leq \mathsf{Adv}^{\mathsf{ind}}_{\pi}(\mathcal{A}[0]) + 2(e_2 + d_2) \left(\begin{array}{l} \mathsf{Adv}^{\mathsf{ind}}_{\pi}(t, e_1 + e_2, d_1 + d_2, 0, 0) \\ +\mathsf{Adv}^{\mathsf{ind}}_{\pi^{-1}}(t, d_1 + d_2 - 1, e_1 + e_2 + 2, 0, 0) \end{array} \right).$$

Car $\mathcal{A}[0]$ est un $(t, e_1, d_1, 0, 0)-\mathsf{IND}$ attaquant dont l'avantage est borné par $\mathsf{Adv}^{\mathsf{ind}}_{\pi}(t, e_1 + e_2, d_1 + d_2, 0, 0)$, on obtient le résultat. □

Dans plusieurs chiffrements par bloc, l'algorithme de chiffrement et l'algorithme de déchiffrement sont très similaires. Par conséquent, si le chiffrement par bloc est sûr contre tout attaquant ($\mathsf{IND}-\mathsf{P1}-\mathsf{C1}$), son inverse atteint un niveau similaire de sécurité. Le théorème ci-dessus implique que le chiffrement par bloc est, dans ce cas, sûr face aux attaquants adaptatifs ($\mathsf{IND}-\mathsf{P2}-\mathsf{C2}$). Les attaques adaptatives ne donnent pas d'avantage supplémentaire contre les chiffrements par bloc.

3.3 Indistinguabilité et pseudo-aléatoire

Dans cette partie, nous introduisons une relation entre la notion d'indistinguabilité définie ci-dessus et la notion de permutations pseudo-aléatoires classiques des chiffrements par bloc.

3.3.1 IND−P1−C0 est équivalente à la propriété de permutations pseudo-aléatoires

Dans [40], Desai et Miner ont affirmé que :

Proposition 32 *Pour tout chiffrement par bloc π :*

$$\frac{1}{2} \times \mathsf{Adv}_{\pi}^{\mathsf{ind}}(t, e_1, 0, 0, 0) \leq \mathsf{Adv}_{\pi}^{\mathsf{prp}}(t, e_1 + 1) \leq (e_1 + 1) \times \mathsf{Adv}_{\pi}^{\mathsf{ind}}(t, e_1 + 1, 0, 0, 0).$$

Nous prouvons cette proposition (dont la démonstration n'avait jamais été publiée et notamment pas dans [40]) grâce aux théorèmes 33 et 34 qui seront énoncés dans les sections suivants, dont la portée est plus large. En effet, la relation de gauche de la proposition 32 est un cas particulier du Théorème 33 où $d_1 = e_2 = d_2 = 0$, tandis que la relation de droite est un cas particulier du Théorème 34 où $n = e_1 + 1$ et $m = 0$. Pour la dernière inégalité, puisque la dernière requête est toujours une requête de chiffrement ($m = 0$ implique qu'il n'y a aucune requête de déchiffrement à la deuxième étape), on ne doit jamais simuler une requête de déchiffrement. Par conséquent, l'avantage par rapport au schéma π^{-1} disparaît (dans notre preuve, seul l'attaquant $\mathcal{B}$ est à construire).

3.3.2 IND−P2−C2 est « presque » équivalente à la propriété de permutations super pseudo-aléatoires

Théorème 33 *Pour tout chiffrement par bloc π :*

$$\mathsf{Adv}_{\pi}^{\mathsf{ind}}(t, e_1, d_1, e_2, d_2) \leq 2 \times \mathsf{Adv}_{\pi}^{\mathsf{sprp}}(t, e_1 + e_2 + 1, d_1 + d_2).$$

Preuve. On suppose que π est SPRP-sûr. On montre alors que π est également sûr face aux IND−P2−C2-attaquants. Considérons $\mathcal{A}$, un (t, e_1, e_2, d_1, d_2)−IND-attaquant, contre le schéma π. L'objectif est de montrer que $\mathsf{Adv}_{\pi}^{\mathsf{ind}}(\mathcal{A})$ est négligeable. Pour ce faire, on construit un SPRP-attaquant $\mathcal{B}$ contre π en utilisant $\mathcal{A}$ comme un sous-programme.

Description de $\mathcal{B}^{\mathcal{O}_b, \mathcal{O}_b^{-1}}$ $\mathcal{B}$ exécute $\mathcal{A}_1$ en répondant aux requêtes de chiffrement / déchiffrement qui sont simplement transmises respectivement aux oracles $\mathcal{O}_b$ et $\mathcal{O}_b^{-1}$. Quand $\mathcal{A}_1$ retourne deux candidats (m_0, m_1) et s, $\mathcal{B}$ choisit aléatoirement un bit d et calcule

$y_d = \mathcal{O}_b(m_d)$. $\mathcal{B}$ continue a exécuter $\mathcal{A}_2(y_d)$, et transmet encore toutes les requêtes de chiffrement / déchiffrement aux oracles $\mathcal{O}_b$ et $\mathcal{O}_b^{-1}$. Quand $\mathcal{A}_2$ devine d' pour d, $\mathcal{B}_2$ vérifie si $d' = d$: $\mathcal{B}_2$ devine $b' = 1$ pour le bit b si oui, il devine $b' = 0$ sinon.

Avantage de $\mathcal{B}$ On considère la relation entre l'avantage de $\mathcal{B}$ et celui de $\mathcal{A}$.

- dans le cas où $b = 1$, le jeu est exactement celui dans lequel $\mathcal{A}$ joue contre π. La probabilité que $\mathcal{B}$ retourne $b' = 1$ est donc la probabilité que $d' = d$: $(\mathsf{Adv}_{\pi}^{\mathsf{ind}}(\mathcal{A}) + 1)/2$.
- dans le cas où $b = 0$, $\mathcal{A}$ accède en fait à une permutation aléatoire, et $y_d = \mathcal{P}(m_d)$ est parfaitement indépendant de m_0 et m_1. Par conséquent, $\mathcal{A}_2$ devine correctement $d' = d$ avec probabilité $1/2$. Ceci entraîne que $\mathcal{B}$ devine $b' = 1$ avec une probabilité $1/2$.

En combinant les deux cas, on obtient le résultat. Remarquons que $\mathcal{B}$ fait $e_1 + e_2 + 1$ requêtes à l'oracle $\mathcal{O}_b$, la requête additionnelle est nécessaire pour calculer y_d. □

L'autre direction est moins naturelle, et donc, plus surprenante :

Théorème 34 *Pour tout chiffrements par bloc π :*

$$\mathsf{Adv}_{\pi}^{\mathsf{sprp}}(t, n, m) \leq (n + m) \times \left(\mathsf{Adv}_{\pi}^{\mathsf{ind}}(t, n, m, 0, 0) + \mathsf{Adv}_{\pi^{-1}}^{\mathsf{ind}}(t, m, n, 0, 0) \right).$$

Preuve. Considérons $\mathcal{A}$, un $(t, n, m)-\mathsf{SPRP}$-attaquant, contre le schéma π. D'après la proposition 26, on peut supposer que $\mathcal{A}$ est un attaquant normal.

On dénote $\mathcal{A}[\eta]$ un nouvel attaquant que l'on construit à partir de $\mathcal{A}$ en restreignant les interactions de $\mathcal{A}$: les η premières requêtes sont transmises à l'oracle $\mathcal{O}$ (s'il s'agit d'une requête de chiffrement) et à l'oracle $\mathcal{O}^{-1}$(s'il s'agit d'une requête de déchiffrement). Les autres requêtes de chiffrement et de déchiffrement ont pour réponse respectivement des permutations aléatoires $\mathcal{P}$ et $\mathcal{P}^{-1}$. L'objectif de l'attaquant est toujours de deviner le bit b'. On définit $\mathsf{PI}(\mathcal{A}[\eta])$ la probabilité que $\mathcal{A}[\eta]$ retourne la réponse $b' = 1$. On a :

$$\begin{aligned} \mathsf{Adv}_{\pi}^{\mathsf{sprp}}(\mathcal{A}) &= \Pr[\mathcal{A}() = 1 \,|\, b = 1] - \Pr[\mathcal{A}() = 1 \,|\, b = 0] \\ &= \Pr[\mathcal{A}^{\mathcal{E}_k, \mathcal{D}_k}() = 1] - \Pr[\mathcal{A}^{\mathcal{P}, \mathcal{P}^{-1}}() = 1] = \mathsf{PI}(\mathcal{A}[n + m]) - \mathsf{PI}(\mathcal{A}[0]). \end{aligned}$$

Lemme 35 *Pour tout $\eta \leq n + m$:*

$$\mathsf{PI}(\mathcal{A}[\eta]) - \mathsf{PI}(\mathcal{A}[\eta - 1]) \leq \mathsf{Adv}_{\pi}^{\mathsf{ind}}(n, m, 0, 0) + \mathsf{Adv}_{\pi^{-1}}^{\mathsf{ind}}(m, n, 0, 0) \tag{3.2}$$

Preuve. La preuve complète est présentée en détail dans [97]. L'idée est assez similaire à celle de la preuve du lemme 31. On construit deux attaquants : $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ un $(t, n, m, 0, 0)-\mathsf{IND}$ attaquant contre π, et $\mathcal{C} = (\mathcal{C}_1, \mathcal{C}_2)$ un $(t, n, m, 0, 0)-\mathsf{IND}$ attaquant contre π^{-1}. Les attaquants sont construits tels que un des avantages est exactement la terme à gauche de la relation 3.2. Ces attaquants exécutent $\mathcal{A}$ jusqu'à la η^{e} requête de

$\mathcal{A}[\eta]$ en utilisant $\mathcal{E}_\mathsf{k}$ pour répondre à une requête à $\mathcal{O}_b$ et $\mathcal{D}_\mathsf{k}$ pour répondre à une requête à $\mathcal{O}_b$. Selon la nature de la η^e requête de $\mathcal{A}[\eta]$ (chiffrement ou déchiffrement), $\mathcal{B}_1$ ou $\mathcal{C}_1$ la choisit comme un des deux candidats (l'autre candidat est aléatoirement choisi). Supposons qu'il s'agisse d'une requête de chiffrement q. Alors, $\mathcal{B}_1$ retourne deux candidats q_0, q_1 où q_0 est aléatoirement choisi et $q_1 = q$.

$\mathcal{B}_2$ reçoit donc un challenge $a = \mathcal{E}_\mathsf{k}(m_d)$ pour un bit aléatoire d qu'il doit deviner. $\mathcal{B}_2$ continue à exécuter $\mathcal{A}$ de la façon suivante : il utilise deux permutations aléatoires $\mathcal{P}$ et $\mathcal{P}^{-1}$ pour répondre respectivement aux requêtes de chiffrement et de déchiffrement de $\mathcal{A}$. Quand $\mathcal{A}$ devine b' pour b, $\mathcal{B}_2$ vérifie si $b' = b$: si oui, $\mathcal{B}_2$ devine $d' = 1$, sinon, il devine $d' = 0$.

On remarque que si a est le chiffrement de q_0, l'attaquant $\mathcal{B}$ est en fait $\mathcal{A}[\eta - 1]$ et si a est le chiffrement de q_1, l'attaquant $\mathcal{B}$ est en fait $\mathcal{A}[\eta]$. On peut donc montrer que l'avantage pour deviner le bit d de $\mathcal{B}$ est la différence entre l'avantage de $\mathcal{A}[\eta]$ de deviner le bit b et celui de $\mathcal{A}[\eta - 1]$ de deviner le bit b. □

En appliquant ce lemme $n + m$ fois, on obtient le résultat. □

À partir des deux théorèmes précédents, nous constatons qu'un chiffrement par bloc π est une famille de permutations super pseudo-aléatoires si et seulement si π et son inverse π^{-1} atteignent la sécurité sémantique face aux attaques non-adaptatives (IND−P1−C1). Autrement dit, sous la conjecture qu'un chiffrement par bloc et son inverse atteignent le même niveau de sécurité face aux attaques non-adaptatives, SPRP et IND−P1−C1 sont équivalentes avec une réduction linéaire.

De plus, nous obtenons une équivalence plus intuitive entre IND−P2−C2 et SPRP sous une condition plus faible : π^{-1} est seulement IND−P1−C0. Ce résultat est présenté dans la version complète de l'article [97].

3.4 Application à DES

D'abord, on remarque que, dans tous les chiffrements par bloc de type Feistel, les algorithmes de chiffrement et de déchiffrement sont « presque » identiques. La seule différence est que le chiffrement utilise la sous-clef K_i pour le tour i (pour $i = 1, \ldots, n$ où n est le nombre de tours), alors que le déchiffrement utilise la sous-clef K_{n+1-i} pour le tour i. Si on fait la conjecture que l'utilisation des sous-clefs dans l'ordre inversé n'influence pas la sécurité du schéma, on peut prouver qu'une attaque adaptative n'est pas plus puissante qu'une attaque non-adaptative, avec une réduction linéaire. Dans le cas de DES, nous montrons que cette conjecture est vraie si le remplacement de la rotation à gauche par une rotation à droite, donc une petite modification de la dérivation des sous-clefs (« key schedule » en anglais), n'affaiblit pas la sécurité du schéma.

Dérivation des sous-clefs du DES On rappelle brièvement la dérivation des sous-clefs du DES que l'on appelle ici *dérivation des sous-clefs à gauche*. Dans cette partie, DES_L désigne le chiffrement par bloc DES.

Entrée : La clef principale K, une chaîne de 56 bits (on ne considère pas les bits de parité) ;

Sortie : Les sous-clefs $K_1, \ldots, K_{16}$;

Processus :

- Étape 0 : Transformation de la clef principale K en $L_0 = (C_0, D_0)$ par la table de permutation PC-1. L_0 est une chaîne de 56 bits (deux parties de 28 bits).

Pour chaque $i = 1, \ldots, 16$:

- Étape i : Transformation de $L_{i-1} = (C_{i-1}, D_{i-1})$ en $L_i = (C_i, D_i)$, où $C_i = \mathsf{LR}_i(C_{i-1})$ et $D_i = \mathsf{LR}_i(D_{i-1})$. Dans cette transformation, LR_i dénote la rotation à gauche de deux positions ou d'une seule position (quand $i = 1, 2, 9, 16$). On dénote LO_i les rotations *à gauche* qui opèrent sur l'état global : $L_i = \mathsf{LO}_i(L_{i-1})$. Transformation de L_i en K_i par la table de permutation PC-2.

Maintenant, on modifie un peu la dérivation des sous-clefs en remplaçant la rotation à gauche (LR) par la rotation à droite (RR). On obtient donc une nouvelle dérivation des sous-clefs que l'on appelle *dérivation des sous-clefs à droite*. On dénote DES_R la variante de DES, où *dérivation des sous-clefs à gauche* est remplacée par *dérivation des sous-clefs à droite* :

Entrée : La clef principale K, une chaîne de 56 bits (on ne considère pas les bits de parité) ;

Sortie : Les sous-clefs $K_1, \ldots, K_{16}$.

Processus :

- Étape 0 : Transformation de la clef principale K en $R_0 = (C_0, D_0)$ par la table de permutation PC-1. R_0 est une chaîne de 56 bits (deux parties de 28 bits).

Pour chaque $i = 1, \ldots, 16$,

- Étape i : Transformation de $R_{i-1} = (C_{i-1}, D_{i-1})$ en $R_i = (C_i, D_i)$, où $C_i = \mathsf{RR}_i(C_{i-1})$ et $D_i = \mathsf{RR}_i(D_{i-1})$. Dans cette transformation, RR_i dénote la rotation à droite de deux positions ou d'une seule position (quand $i = 1, 2, 9, 16$). On dénote RO_i les rotations *à droite* qui opèrent sur l'état global : $R_i = \mathsf{RO}_i(R_{i-1})$. Transformation de R_i en K_i par la table de permutation PC-2.

Remarquons que $\mathsf{LO}_i = \mathsf{RO}_{18-i}^{-1}$ pour $i = 2, \ldots, 16$, et $\mathsf{LO}_1 = \mathsf{RO}_1^{-1}$.

La petite modification de dérivation des sous-clefs qui remplace la rotation à gauche par celle à droite n'influence probablement pas la sécurité du schéma. On fait alors la conjecture suivante :

Conjecture 36 (Conjecture gauche-droite de DES) *Les deux chiffrements par bloc* DES_L *et* DES_R *atteignent le même niveau de sécurité.*

Cependant, pour le résultat suivant, on a besoin d'une hypothèse plus faible :

Conjecture 37 (Conjecture gauche-droite de DES non-adaptative) *Les deux chiffrement par bloc* DES_L *et* DES_R *atteignent le même niveau de sécurité face aux attaques non-adaptatives :*

$$\left|\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{DES}_L}(t, e_1, d_1, 0, 0) - \mathsf{Adv}^{\mathsf{ind}}_{\mathsf{DES}_R}(t, e_1, d_1, 0, 0)\right|$$

est faible.

Théorème 38 *Sous la* conjecture gauche-droite de DES non-adaptative*, un attaquant adaptatif n'a pas plus de puissance qu'un attaquant non-adaptatif pour casser* DES *au sens de la sécurité sémantique :*

$$\left|\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{DES}}(t, e_1, d_1, e_2, d_2) - (2(e_2 + d_2) + 1)\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{DES}}(t, q, q, 0, 0)\right|$$

est négligeable, où $q = \max\{e_1 + e_2, d_1 + d_2\}$.

Preuve. Il suffit de montrer que sous la *conjecture gauche-droite de* DES *non-adaptative*,

$$\left|\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{DES}}(t, e_1, d_1, 0, 0) - \mathsf{Adv}^{\mathsf{ind}}_{\mathsf{DES}^{-1}}(t, e_1, d_1, 0, 0)\right| \leq \delta.$$

Considérons le processus de *dérivation des sous-clefs à gauche* qui commence avec $L_0 = \mathsf{PC\text{-}1}(K)$, puis $L_i = \mathsf{LO}_i(L_{i-1})$ et $K_i = \mathsf{PC\text{-}2}(L_i)$. Remarquons que $L_{16} = L_0$ (car on a 16 rotations, dont douze sont à deux positions et quatre à une seule position, qui opèrent de façon indépendante sur chaque partie de 28 bits).

Considérons le processus de *dérivation des sous-clefs à droite* qui commence avec $R_0 = L_1$ (qui est donc $\mathsf{PC\text{-}1}(K')$ pour une clef K', car PC-1 est une permutation). Alors $R_i = \mathsf{RO}_i(R_{i-1})$ et $K'_i = \mathsf{PC\text{-}2}(R_i)$. De façon similaire, comme le cas précédent, $R_{16} = R_0 = L_1$:

$$L_1 = R_{16} = \mathsf{RO}_{16}(R_{15}), \text{ qui implique que } R_{15} = \mathsf{RO}^{-1}_{16}(L_1) = \mathsf{LO}_2(L_1) = L_2.$$

Par induction, si $R_{17-i} = L_i$ (ce qui est vrai pour $i = 1, 2$),

$$L_i = R_{17-i} = \mathsf{RO}_{17-i}(R_{17-(i+1)}), \text{ et, } R_{17-(i+1)} = \mathsf{RO}^{-1}_{17-i}(L_i) = \mathsf{LO}_{i+1}(L_i) = L_{i+1}.$$

Par conséquent, pour $i = 1, \ldots, 16$, $L_i = R_{17-i}$ et $K_i = K'_{17-i}$, où $K' = \mathsf{PC\text{-}1}^{-1}(R_0)$:

$$\mathsf{DES}^{-1}_L(K) = \mathsf{DES}_R(K'), \text{ pour } K' = \mathsf{PC\text{-}1}^{-1}(\mathsf{PC\text{-}2}(\mathsf{PC\text{-}1}(K))).$$

Comme PC-1 et PC-2 sont des permutations, quand K est uniformément distribuée, K' l'est aussi : DES_R et $\mathsf{DES}^{-1}_L = \mathsf{DES}^{-1}$ sont donc parfaitement équivalentes en terme de sécurité. En conclusion, sous la *conjecture gauche-droite de* DES *non-adaptative*, DES et DES^{-1} atteignent le même niveau de sécurité face aux attaquants non-adaptatifs, d'après le théorème 30. □

3.5 Conclusion

La plupart des chiffrements par bloc sont construits tels que les algorithmes de chiffrement et de déchiffrement sont similaires (le code est plus court). Intuitivement, ils devraient avoir le même niveau de sécurité. À titre d'exemple, nous avons démontré que ceci est vrai pour DES, l'un des plus célèbres chiffrements par bloc, sous une faible conjecture (conjecture 36).

Plus précisément, nous avons démontré que, les attaques adaptatives ne sont pas significativement plus puissantes que les attaques non adaptatives. Nous prouvons également que la notion de permutations super pseudo-aléatoires (qui est la plus utilisée pour les chiffrements par bloc) est équivalente à la sécurité sémantique face aux attaques adaptatives. Par transitivité, cette équivalence est aussi valable pour les attaques non-adaptatives.

Construction de nouveaux schémas de chiffrement asymétrique

4

Introduction

Dans la mise en œuvre des schémas de chiffrement, l'efficacité est un des facteurs les plus importants. Depuis longtemps, on essaie de construire des schémas de chiffrement à la fois efficaces et sûrs. Malheureusement, la sécurité forte va rarement de pair avec la mise en œuvre pratique. Un compromis doit être fait et en général, on fait quelques hypothèses sur l'attaque. À titre d'exemple, elle pourrait être générique et indépendante de l'exécution réelle de certains objets tels que les fonctions de hachage (dans le modèle de l'oracle aléatoire [49, 9]), le chiffrement symétrique par bloc (dans le modèle du « chiffrement idéal ») ou les groupes algébriques (dans le modèle générique [27]). Parmi ces hypothèses idéalistes, celle sur les fonctions de hachage — le modèle de l'oracle aléatoire — est la plus étudiée. En effet, nombre de schémas cryptographiques utilisent des fonctions de hachage telles que MD5 [105], les normes américaines SHA-1 [90], SHA-256, SHA-384 et SHA-512 [91].

Dans le « modèle de l'oracle aléatoire », introduit par Bellare et Rogaway [9], la fonction de hachage est formalisée comme un oracle qui retourne une valeur parfaitement aléatoire, sous réserve qu'elle produise toujours le même résultat pour les entrées identiques. Ce modèle est une passerelle entre la cryptographie théorique et la cryptographie pratique : le schéma est d'abord prouvé sûr dans le modèle de l'oracle aléatoire, les oracles sont remplacés par des fonctions réelles de hachage convenablement choisies. Bien qu'elle ne fournisse que des preuves de sécurité idéalisées, cette méthode a permis la mise au point de schémas particulièrement efficaces comme OAEP (*Optimal Asymmetric Encryption Padding*) [10] pour le chiffrement ou PSS (*Probabilistic Signature Scheme*) [11, 35] pour la signature.

Dans la littérature, plusieurs séparations entre le modèle standard et celui de l'oracle aléatoire ont déjà été montrées [29, 88, 5, 62, 30] (quelques unes ont également été montrées par rapport au modèle générique [116]). Cependant, les contre-exemples sont artificiels et contre-intuitifs ; ils ne sont pas applicables aux scénarios réels. De plus, une preuve de sécurité dans le modèle de l'oracle aléatoire sous une hypothèse faible peut apporter plus d'intérêt pratique qu'une preuve de sécurité dans le modèle standard sous une hypothèse forte. Le modèle de l'oracle aléatoire est donc encore largement accepté.

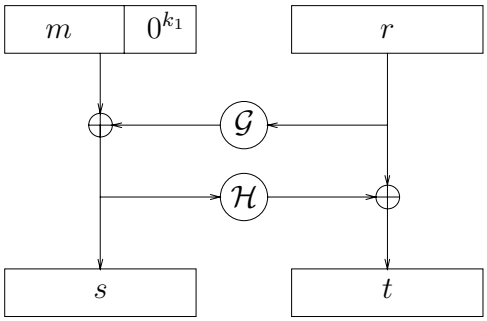

FIG. 4.1 – Optimal Asymmetric Encryption Padding

Dans cette partie, nous mettons l'accent sur l'efficacité des schémas de chiffrement et considérons la construction la plus célèbre du chiffrement à clef publique : la construction OAEP, introduite par Bellare et Rogaway [10].

4.1 OAEP

Avec l'introduction de la notion formelle du modèle de l'oracle aléatoire en 1993, Bellare et Rogaway ont proposé un schéma efficace de chiffrement [10] pour optimiser l'efficacité. Cet objectif entraîne deux problèmes de minimisation : l'un concerne l'écart entre la taille du texte chiffré et celle du texte clair (*i.e.* la bande passante) et l'autre, la taille du chiffré par rapport à celle de la primitive utilisée. Leur construction, appelée OAEP, résout le deuxième problème d'optimisation : avec une permutation à sens-unique à trappe de n bits vers n bits, la longueur du texte chiffré peut être optimale, *i.e.* n bits. Cette construction, illustrée dans la figure 4.1, permet de convertir une permutation à sens-unique à trappe en un schéma de chiffrement [10] décrit comme suit :

4.1.1 Description

Notations et paramètres communs

Le chiffrement et le déchiffrement utilisent deux fonctions de hachage : $\mathcal{G}$, $\mathcal{H}$, modélisées par des oracles aléatoires dans les analyses de sécurité. Les paramètres de sécurité satisfont $n = k + \ell$:

$$\mathcal{G} : \{0,1\}^k \to \{0,1\}^\ell \qquad \mathcal{H} : \{0,1\}^\ell \to \{0,1\}^k.$$

Le chiffrement utilise une famille de permutations à sens-unique (φ_{pk}) (pour simplifier les notations, la deuxième indice pk signifie l'espace des clef publiques) dont les inverses sont respectivement les ψ_{sk}, où sk est la clef secrète (*i.e.* la trappe) associée à la clef publique

pk. Le symbole « ‖ » dénote la concaténation des chaînes de bits. De plus, on identifie $\{0,1\}^k \times \{0,1\}^\ell$ à $\{0,1\}^n$.

Algorithme de chiffrement

L'espace des textes clairs est $\mathcal{M} = \{0,1\}^{\ell-k_1}$. Le chiffrement utilise un aléa $r \in \mathcal{R} = \{0,1\}^k$ et retourne un texte chiffré c dans $\mathrm{F} = \{0,1\}^n$: à partir du texte clair $m \in \mathcal{M}$, on calcule :

$$s = (m\|0^{k_1}) \oplus \mathcal{G}(r) \qquad t = r \oplus \mathcal{H}(s).$$

Le texte chiffré correspondant est alors $c = \varphi_{\mathsf{pk}}(s,t)$.

Algorithme de déchiffrement

A partir du texte chiffré c, on calcule d'abord $s\|t = \psi_{\mathsf{sk}}(c)$, où $s \in \{0,1\}^k$ et $t \in \{0,1\}^\ell$, puis

$$r = t \oplus \mathcal{H}(s) \quad M = s \oplus \mathcal{G}(r).$$

Si les k derniers bits de M sont tous 0, le déchiffrement retourne le message m qui est M sans les k derniers bits. Dans le cas contraire, le déchiffrement retourne « chiffré non valable », *i.e.* le texte chiffré ne correspond à aucun texte clair.

L'instantiation du schéma avec la fonction RSA [107], appelée RSA-OAEP, devient la norme pour le chiffrement RSA (PKCS#1 version 2, IEEE P1363). Ce schéma est très efficace en termes de temps de calcul et de l'expansion de message, il est aussi compatible avec des applications plus traditionnelles du chiffrement RSA. Outre son efficacité, la raison déterminante de la popularité de f-OAEP fut la confiance en sa sécurité : sémantiquement sûr contre des attaques à chiffrés choisis adaptatives, sous réserve que la permutation utilisée soit à sens-unique à trappe—malgré aucune preuve complète.

Cependant, en 2001, Shoup [114] a montré que pour ce schéma, la sécurité ne peut reposer sur l'hypothèse générale de l'existence de permutations à sens-unique à trappe. En effet, il a introduit une nouvelle technique, appelée la technique des « jeux successifs », pour prouver la sécurité. Appliquée au schéma OAEP, Shoup s'est heurté à un obstacle insurmontable. Il en a alors déduit un contre-exemple pour mettre en évidence le problème de sécurité d'OAEP. Cette technique sera étudiée en détail dans la section 4.2

4.1.2 Attaque de Shoup

Tout d'abord, on rappelle les étapes pour casser un schéma au sens de la sécurité sémantique : l'attaquant choisit deux messages m_0 et m_1 ; reçoit un challenge c-le chiffré d'un de ces deux messages ; et finalement, devine à quel message c correspond. Dans le jeu d'attaque CCA2, on peut accéder à l'oracle de déchiffrement à n'importe quel moment. L'idée de Shoup est décrite comme suit :

Quand on reçoit le challenge c, on vise à produire un autre texte chiffré c' tel que les valeurs r et r' dans la construction de ces deux chiffrés soient égales. Dans le cas où $r = r'$, on obtient une relation entre les deux texte clairs : $m \oplus m' = s \oplus s'$. Par conséquent, si on connaît s, s', grâce à une seule requête de déchiffrement sur le texte chiffré c', on obtient m', puis le m correspondant au challenge c. Le problème restant est de comprendre la façon avec laquelle un tel texte chiffré c' est généré. Pour cela, Shoup a introduit la notion de permutation à sens-unique à trappe « XOR-malléable » : pour une telle permutation f_0, on peut, avec probabilité non-négligeable, calculer $f_0(t \oplus a)$ à partir de $f_0(t)$ et a.

Supposons maintenant que l'on dispose d'une permutation f_0 à sens-unique à trappe « XOR-malléable ». Définissons f par $f(s||t) = s||f_0(t)$. Il est facile de montrer que f est également une permutation à sens-unique à trappe. Alors, du challenge $c = f(s||t) = s||f_0(t)$, on extrait directement s. On choisit aléatoirement s' et calcule $a = \mathcal{H}(s) \oplus \mathcal{H}(s')$. À partir de a, $f_0(t)$, grâce à la propriété de permutation XOR-malléable, l'attaquant peut calculer $f_0(t' = t \oplus \mathcal{H}(s) \oplus \mathcal{H}(s'))$. Considérons la valeur r' du texte chiffré $c' = s'||f_0(t') : r' = t' \oplus \mathcal{H}(s') = t \oplus \mathcal{H}(s) \oplus \mathcal{H}(s') \oplus \mathcal{H}(s') = t \oplus \mathcal{H}(s) = r$. Par conséquent, par le même raisonnement, une fois que c' est demandé à l'oracle de déchiffrement, on peut calculer le texte clair m du m' retourné par l'oracle de déchiffrement.

Dans son article, Shoup a construit un modèle de calcul non standard où il existe une permutation à sens-unique à trappe XOR-malléable. Par conséquent, il est impossible de formuler une preuve de la propriété $\mathsf{IND-CCA2}$ sûr de f-OAEP en utilisant la fonction f comme une boîte noire. f-OAEP ne pourrait être $\mathsf{IND-CCA2}$ sûr qu'avec des propriétés spécifiques de la fonction f.

4.1.3 Sécurité de RSA-OAEP

Heureusement, OAEP est presque immédiatement sauvé par Fujisaki *et al.* [54]. Ces auteurs ont montré que : premièrement, l'utilisation d'une permutation à « sens-unique partielle » [3] à trappe dans OAEP garantit son niveau fort de sécurité ; deuxièmement, la fonction RSA est une permutation à sens-unique partielle à trappe sous l'hypothèse bien connue qu'elle soit simplement une permutation à sens-unique à trappe. La sécurité de RSA-OAEP est définitivement prouvée. Cependant, la réduction dans leur preuve n'est pas aussi bonne qu'espérée. En effet, le coût de la réduction du problème d'inverser partiellement la fonction RSA au problème de sécurité de RSA-OAEP est quadratique en temps. De plus, le coût de la réduction du problème d'inverser complètement la fonction RSA au problème d'inverser partiellement la fonction RSA est aussi quadratique en temps.

[3] Une fonction f est k-partiellement à sens-unique si, à partir de $f(x)$, il est calculatoirement difficile de trouver les k premiers bits de x.

4.2 Technique des jeux successifs

Au cours du temps, l'exigence du niveau de sécurité grandit. Les notions de sécurité deviennent ainsi de plus en plus complexes. Par conséquent, il est de plus en plus difficile de prouver la sécurité d'un schéma et de la vérifier. À titre d'exemple, la construction OAEP a été utilisée pendant une longue période sans être ni prouvée sûre ni attaquée. Dans ce contexte, Shoup a introduit la technique des « jeux successifs » pour une construction progressive des preuves de sécurité. Chaque étape est donc simple et facile à vérifier.

4.2.1 Description

La sécurité d'un schéma peut être évaluée au travers d'un « jeu » d'attaque entre un « attaquant » et un « challengeur ». L'attaquant et le challengeur sont des algorithmes probabilistes interactifs. Ils génèrent ainsi un espace probabiliste. Normalement, la définition de sécurité est liée à un « événement » particulier Ev : l'attaquant casse le schéma. La sécurité exige que, pour tout attaquant, la probabilité d'occurrence de cet événement soit « très proche » d'une valeur « cible », typiquement 0 ou $\frac{1}{2}$. La technique de jeux successifs consiste à changer petit à petit, pour chaque jeu, le comportement du challengeur. Le changement est effectué de telle manière que, entre deux jeux successifs, la différence de probabilité d'occurrence de l'événement Ev est « négligeable ». L'objectif est d'obtenir, dans le jeu final, une estimation de la probabilité d'occurrence de l'événement Ev. L'un des avantages de cette technique est que l'on peut facilement vérifier la preuve grâce à la simplicité des transitions des jeux successifs.

Les transitions entre deux jeux successifs les plus utilisées sont au nombre de trois :

- **Transition 1 : Transition basée sur l'indistinguabilité**. Un attaquant qui détecte le changement peut être transformé en un algorithme efficace, capable de distinguer deux distributions indistinguables (statistiquement ou calculatoirement).
- **Transition 2 : Transition basée sur un événements d'échec**. Les i-ème et $(i+1)$-ème jeux procèdent de façon identique sauf si un événement d'échec Ech se produit. Autrement dit :

  $$\Pr[\mathsf{Ev}_i \wedge \neg\mathsf{Ech}] = \Pr[\mathsf{Ev}_{i+1} \wedge \neg\mathsf{Ech}].$$

  On constate que l'écart entre les probabilités d'occurrence des événements Ev_i et Ev_{i+1} dans les deux jeux successifs est, elle-même, « négligeable » si la probabilité d'occurrence de l'événement d'échec Ech est « négligeable ». Ceci résulte du lemme suivant (dont la preuve est évidente) :

  Lemme 39 *Soient* E, F *et* G *des événements dans un espace de probabilités, alors*

  $$\Pr[\mathsf{E} \wedge \neg\mathsf{G}] = \Pr[\mathsf{F} \wedge \neg\mathsf{G}] \implies |\Pr[\mathsf{E}] - \Pr[\mathsf{F}]| \leq \Pr[\mathsf{G}].$$

- **Transition 3 : Transition de ré-écriture**. Il s'agit d'une étape intermédiaire, qui décrit les façons équivalentes de calculer certaines quantités dans les jeux successifs. Cette transition semble inutile mais elle permet de suivre la preuve plus facilement.

L'état de l'art de cette technique est dans [115]

4.2.2 Exemple

Dans [9], outre la formalisation de la notion de l'oracle aléatoire, Bellare et Rogaway ont proposé une construction générique de chiffrement $\mathsf{IND}-\mathsf{CCA2}$ à partir de n'importe quelle permutation à sens-unique à trappe.

Cette construction fait appel à deux oracles aléatoires $\mathcal{G}$ et $\mathcal{H}$, à valeurs respectivement dans $\{0,1\}^k$ et $\{0,1\}^\ell$. L'algorithme de génération des clés définit une permutation φ_{pk} dans l'espace E de taille n bits, calculable grâce à la clé publique pk. Son inverse ψ_{sk} ne peut être calculée que grâce à la clef privée sk (la trappe). Pour chiffrer un message $m \in \{0,1\}^k$, on choisit $r \stackrel{R}{\leftarrow} E$, puis calcule :

$$\mathcal{E}(m;r) = \varphi_{\mathsf{pk}}(r) \,\|\, m \oplus \mathcal{G}(r) \,\|\, \mathcal{H}(m,r).$$

Le déchiffrement d'un chiffré $C = a \,\|\, b \,\|\, c$ se fait en deux étapes : tout d'abord, on retrouve $r = \psi_{\mathsf{sk}}(a)$, grâce à la trappe sk, puis $m = b \oplus \mathcal{G}(r)$; ensuite, avant de retourner le message m, on vérifie la consistance du chiffré, *i.e.* si $c = \mathcal{H}(m,r)$.

Ce schéma est prouvé $\mathsf{IND}-\mathsf{CCA2}$ sûr sous l'hypothèse que la famille des permutation (φ_{pk}) φ_{pk} est à sens-unique.

Théorème 40 *[102] Considérons un attaquant $\mathcal{A}$ à chiffrés choisis adaptatif. Supposons qu'après q_D requêtes à l'oracle de déchiffrement et q_G, q_H requêtes aux oracles $\mathcal{G}$ et $\mathcal{H}$, $\mathcal{A}$ a un avantage ε en temps t, alors on peut inverser φ avec succès $\varepsilon/2 - q_D/2^\ell$, en temps $t + (q_G + q_H)T_\varphi$, où T_φ désigne le temps pour une évaluation d'une fonction φ_{pk}.*

Une preuve utilisant la technique des jeux successifs est présentée dans [102]. On va l'étudier plus en détails pour apprécier l'efficacité de cette technique.

Preuve. Considérons l'attaquant $\mathcal{A} = (A_1, A_2)$ contre ce schéma. Dans les deux étapes, A_1 et A_2 ont accès à l'oracle de déchiffrement.

JEU $\mathbf{J}_0$: Il s'agit d'une simulation parfaite du comportement du challengeur. On exécute l'algorithme de génération de clés qui retourne une permutation φ_{pk} et son inverse ψ_{sk}. On génère également $x \stackrel{R}{\leftarrow} E$ et $y = \varphi_{\mathsf{pk}}(x)$. Après avoir vu la clé publique (la description de la fonction φ_{pk}), A_1 retourne deux messages m_0 et m_1. Après avoir reçu le chiffré $C^\star = a^\star \,\|\, b^\star \,\|\, c^\star$ du message m_δ, A_2 retourne un bit δ'. On dénote $r^\star$ l'unique élément tel que $C^\star = \mathcal{E}(m_\delta, r^\star)$. Avec probabilité $(\varepsilon+1)/2$, $\delta' = \delta$. On note cet événement S_0, ainsi que S_i dans les jeux Jeu_i ci-dessous : $\Pr[S_0] = (1+\varepsilon)/2$.

Pour la suite, on pourra supposer que toute question $\mathcal{H}(\star, \rho)$ est précédée de la question $\mathcal{G}(\rho)$, ainsi $q'_G = q_H + q_G$, où q'_G est le nombre total de requêtes à l'oracle $\mathcal{G}$.

JEU $\mathbf{J}_1$: dans un premier temps, on remplace les oracles $\mathcal{G}$ et $\mathcal{H}$ par des simulations classiques : pour toute nouvelle question à l'un de ces oracles, on répond par une chaîne

aléatoire dans l'espace correspondant, puis on stocke les questions-réponses respectivement dans les listes $\mathcal{G}$-List et $\mathcal{H}$-List. Il s'agit de simulations parfaites, $\Pr[S_1] = \Pr[S_0]$.

Commentaire. *Il s'agit d'une transition de type 1 : la simulation des oracles aléatoires est parfaite, les deux distributions sont parfaitement identiques.*

JEU $\mathbf{J}_2$: dans ce jeu, on simule l'oracle de déchiffrement. À la question $C = a \,\|\, b \,\|\, c$, pour $a = f(r)$, si r n'est pas dans $\mathcal{G}$-List, on rejette le chiffré ; si de même $(b \oplus G(r), r)$ n'est à son tour pas dans $\mathcal{H}$-List, on rejette le chiffré ; dans les autres cas, on continue à utiliser l'oracle de déchiffrement.

Avec l'hypothèse ci-dessus que toute question $\mathcal{H}(\star, \rho)$ est précédée de la question $\mathcal{G}(\rho)$, on ne peut refuser un chiffré valide que si $(b \oplus \mathcal{G}(r), r)$ n'a pas été demandé à $\mathcal{H}$. Mais alors, $\mathcal{H}(b \oplus \mathcal{G}(r), r)$ retourne un élément parfaitement aléatoire, qui est égal à c avec probabilité $1/2^{k_1}$. Ainsi, $|\Pr[S_2] - \Pr[S_1]| \leq q_D/2^{k_1}$.

Commentaire. *Il s'agit d'une transition de type 2 : l'événement d'échec* Ech *est « r n'est pas dans $\mathcal{G}$-List ou $(b \oplus \mathcal{G}(r), r)$ n'est pas dans $\mathcal{H}$-List ». À l'exception de cet événement, les deux jeux sont identiques.*

JEU $\mathbf{J}_3$: on poursuit la simulation de l'oracle de déchiffrement, sur $C = a \,\|\, b \,\|\, c$, pour $a = f(r)$. On sait que $r \in \mathcal{G}$-List et $(b \oplus \mathcal{G}(r), r) \in \mathcal{H}$-List. On peut alors trouver ce r (en testant si $\varphi_{\mathsf{pk}}(r) = a$ sur toutes les questions à $\mathcal{G}$, grâce à la propriété de permutation de φ_{pk}), puis déchiffrer correctement : $\Pr[S_3] = \Pr[S_2]$.

Commentaire. *Il s'agit d'une transition de type 3 : le jeu $\mathbf{J}_3$ est identique au jeu 2 mais m est calculé de manière différente. En effet, dans le jeu 2 , $r = f^{-1}(a)$, alors que dans le jeu $\mathbf{J}_3$, r sera extrait de $\mathcal{G}$-List (le cas où r n'est pas dans la liste est déjà exclu).*

JEU $\mathbf{J}_4$: dans ce jeu, on définit $a^\star = y = f(x)$, $b^\star = m_\delta \oplus g^+$ et $c = h^+$, où x, g^+ et h^+ sont aléatoires. De plus, à la question $\mathcal{G}(x)$, on répond g^+, et à la question $\mathcal{H}(m_\delta, x)$ on répond h^+. Il s'agit simplement de spécifier certaines valeurs de $\mathcal{G}$ et $\mathcal{H}$, par des valeurs aléatoires, on ne modifie donc pas les distributions : $\Pr[S_4] = \Pr[S_3]$.

Commentaire. *Il s'agit évidemment d'une transition de type 3. Cette transition est importante dans la mesure où l'instance x est insérée dans la génération du challenge.*

JEU $\mathbf{J}_5$: maintenant, on supprime les modifications locales de $\mathcal{G}$ et $\mathcal{H}$. Les réponses aux question $\mathcal{G}(x)$ et $\mathcal{H}(m_\delta, x)$ sont indépendantes de x et m_δ. La seule différence apparaît si l'événement « x a été demandé », nommé AskG (impliqué aussi par la requête à $\mathcal{H}$), a lieu : $|\Pr[S_5] - \Pr[S_4]| \leq \Pr[\mathsf{AskG}]$. Cependant, dans ce dernier jeu, δ est indépendant de la vue de l'attaquant, ainsi $\Pr[S_5] = 1/2$.

L'inégalité triangulaire nous donne :

$$\frac{\varepsilon}{2} = \frac{1+\varepsilon}{2} - \frac{1}{2} = |\Pr[S_0] - \Pr[S_5]| \leq \frac{q_D}{2^{k_1}} + \Pr[\mathsf{AskG}].$$

Commentaire. *Il s'agit d'une transition de type 2. L'événement d'échec est « x a été demandé »,* i.e. AskG. *Il nous reste à montrer que la probabilité d'occurrence de cet événement est « négligeable ». Cependant, l'événement* AskG *permet d'inverser* $\varphi_{\mathsf{pk}}(x)$ *en testant toutes les questions posées à* $\mathcal{G}$*, d'où le résultat.*

□

Cette construction est très efficace par rapport aux autres schémas dans le modèle standard. Cependant, elle n'est pas optimale car la taille du texte chiffré est encore longue, *i.e.* $n + k + \ell$ bits, par rapport à k bits du texte clair. Cette remarque est la motivation de OAEP.

5

Sécurité à chiffrés choisis sans redondance

Sommaire

En étudiant les limites des schémas précédents, nous proposons deux nouveaux schémas de chiffrement : le premier, dont l'efficacité est « optimale », est dans le modèle de la permutation aléatoire et le deuxième, dans le modèle de l'oracle aléatoire. Ce dernier, appelé f-OAEP 3 tours, dispose des propriétés importantes suivantes :

Sécurité : sous l'hypothèse d'une permutation f à sens-unique à trappe, ce schéma est sémantiquement sûr face aux attaques à chiffrés choisis adaptatives.

Sans redondance : tout chiffré correspond au chiffrement d'un texte clair. Il s'agit du premier schéma IND−CCA2 sans redondance. Cette propriété améliore l'efficacité du schéma en optimisant la bande passante.

Flexibilité : f n'est pas nécessairement une permutation mais peut être une fonction satisfaisant quelques conditions spécifiques. Quelques exemples de fonctions admissibles sont le chiffrement ElGamal ou celui de Paillier, d'où les nouvelles variantes d'OAEP 3 tours : ElGamal-OAEP 3 tours et Paillier-OAEP 3 tours. Rappelons que le chiffrement ElGamal ou celui de Paillier ne sont pas applicables aux précédentes variantes d'OAEP.

5.1 Chiffrement sans redondance

Dans le schéma OAEP ainsi que dans les variantes dont la primitive de base est une permutation à sens-unique à trappe (SAEP, SAEP+ [19] et OAEP+ [114]), le texte clair est toujours complété avec de la redondance avant d'être chiffré.

D'autres paddings, applicables à des familles de fonctions plus générales, ont été proposés par Fujisaki et Okamoto [53, 52], Pointcheval [100] et Okamoto et Pointcheval [93]. La sécurité y est également garantie par de la redondance, mais dans ce cas, la redondance concerne le texte chiffré : avant de retourner le texte chiffré, on lui ajoute de la redondance.

L'introduction de redondance dans le texte clair ou dans le texte chiffré a toujours le même objectif : un texte chiffré qui n'est pas proprement généré à partir d'un texte clair n'est valide qu'avec une probabilité négligeable. Cette propriété est formellement définie par la notion de *plaintext-awareness* dans [10, 7]. Elle entraîne le fait que l'oracle de déchiffrement ne donne aucune information aux attaquants.

Ainsi, la redondance renforce la sécurité. Cependant, dans certains cas, elle est source d'attaques. En effet, le mécanisme de déchiffrement pour les textes chiffrés valides pourrait être différent de celui pour les textes chiffrés invalides : après un test de validité, l'algorithme de déchiffrement continue la phase de calcul si le texte chiffré vérifie la forme de redondance, sinon il retourne immédiatement la réponse que le texte chiffré est invalide. Par conséquent, des attaques peuvent être menées [13, 79].

A ce stade, le problème qui se pose est donc d'éviter ces redondances dans le chiffrement. Ceci éviterait d'éventuelles attaques et, surtout, améliorerait l'efficacité du schéma.

5.2 FDP : permutation sur un domaine complet

Dans le même esprit que Full-Domain Hash signature [11, 34], nous proposons un chiffrement dans un nouveau modèle appelé permutation sur un domaine complet (Full-Domain Permutation). Dans ce modèle, nous appliquons d'abord une permutation aléatoire sur le couple texte clair - chaîne aléatoire, le résultat est ensuite chiffré avec une permutation à sens-unique à trappe. Cette construction conduit au premier schéma de

chiffrement $\mathsf{IND}-\mathsf{CCA2}$ sûr, à bande passante optimale et sans redondance, *i.e.* tout chiffré est valide.

5.2.1 Description

Le chiffrement FDP est doté d'une grande efficacité grâce à l'utilisation d'une permutation aléatoire $\mathcal{P}$ (qui est un oracle aléatoire bijectif ou un chiffrement idéal avec une clef particulière, 0 par exemple, voir aussi [108]). L'algorithme de génération de clef sélectionne une permutation à sens-unique à trappe φ_{pk} (son inverse ψ_{sk}, calculable à l'aide de la trappe sk) sur $\{0,1\}^{\ell+k}$, et une permutation aléatoire $\mathcal{P}$ sur le même espace $\{0,1\}^{\ell}\times\{0,1\}^{k}$ qui est identifié à $\{0,1\}^{\ell+k}$. La clef publique définit la permutation φ_{pk}, alors que la clef secrète sk définit l'inverse ψ_{sk} de φ_{pk}. Ainsi,

$$\mathcal{E}_{\mathsf{pk}}(m;r)=\varphi_{\mathsf{pk}}(\mathcal{P}(m,r)) \qquad \mathcal{D}_{\mathsf{sk}}(c)=m,\ \text{où}\ (m,r)=\mathcal{P}^{-1}(\psi_{\mathsf{sk}}(c)).$$

L'espace des textes clairs est $\{0,1\}^{\ell}$, alors que celui des aléas r est $\{0,1\}^{k}$. Remarquons que $\mathcal{P}$ et $\mathcal{P}^{-1}$ sont des permutations publiques.

5.2.2 Résultat de sécurité

Les avantages de ce schéma sont au nombre de trois :
- Le premier, comme déjà mentionné, est la validité de tout texte chiffré : n'importe quel élément de l'espace d'arrivée de l'algorithme de chiffrement peut être atteint par cet algorithme.
- Le deuxième découle du résultat de sécurité dans le théorème 41 ci-dessous : il atteint la sécurité $\mathsf{IND}-\mathsf{CCA2}$ sous la seule hypothèse de la difficulté d'inverser φ.
- Le troisième vient de l'optimalité de la bande passante : pour un niveau de sécurité 2^k, on n'a besoin que d'une chaîne aléatoire de k bits. Évidemment, cette remarque est applicable uniquement au cas général où $\ell \geq k$ (*e.g.*, $k=80$ et $k+\ell=1024$).

Théorème 41 *Considérons un attaquant $\mathcal{A}$ contre φ-FDP, selon une attaque à chiffrés choisis adaptative. Supposons qu'après q_p et q_d requêtes respectivement aux oracles de permutation et de déchiffrement, $\mathcal{A}$ peut avoir un avantage ϵ en temps τ, alors on peut inverser φ avec succès $\mathsf{Succ}^{\mathsf{ow}}_{\varphi}(\tau+2q_p\times T_\varphi)$ en temps $\tau+2q_p\times T_\varphi$, où T_φ est le temps nécessaire pour une évaluation d'une fonction de φ, où :*

$$\epsilon \leq 2\times \mathsf{Succ}^{\mathsf{ow}}_{\varphi}(\tau+2q_p\times T_\varphi)+2\times\left(\frac{(q_p+q_d+1)^2}{2^{k+\ell}}+\frac{q_p}{2^k}+\frac{(q_d+1)^2}{2^{\ell}}\right).$$

Rappelons brièvement que pour tout algorithme $\mathcal{A}$:

$$\mathsf{Succ}^{\mathsf{ow}}_{\varphi}(\mathcal{A})=\Pr_{\substack{\mathsf{pk},\\ x\in\{0,1\}^{k+\ell}}}[\mathcal{A}(\mathsf{pk},\varphi_{\mathsf{pk}}(x))=x],\ \text{et}\ \mathsf{Succ}^{\mathsf{ow}}_{\varphi}(\tau)=\max_{|\mathcal{A}|\leq\tau}\left\{\mathsf{Succ}^{\mathsf{ow}}_{\varphi}(\mathcal{A})\right\}.$$

Oracle $\mathcal{P}$

Une requête $\mathcal{P}(m, r)$ aura une réponse p, où

►**Règle** $\mathsf{EvalP}^{(5)}$

| $p = P(m, r)$.

En outre, si (m, r) est une requête directe de l'attaquant à $\mathcal{P}$, on ajoute $(m, r, p, \varphi_{\mathsf{pk}}(p))$ à $\mathcal{P}$-List.

Oracle $\mathcal{P}^{-1}$

Une requête $\mathcal{P}^{-1}(p)$ aura une réponse (m, r), où

►**Règle** $\mathsf{InvP}^{(5)}$

| $(m, r) = P^{-1}(p)$.

En outre, si p est une requête directe de l'attaquant à $\mathcal{P}^{-1}$, on ajoute $(m, r, p, \varphi_{\mathsf{pk}}(p))$ à $\mathcal{P}$-List.

Oracle $\mathcal{D}$

Une requête $\mathcal{D}_{\mathsf{sk}}(c)$ aura une réponse m, où

►**Règle** $\mathsf{Decrypt}^{(5)}$

| $p = \psi_{\mathsf{sk}}(c)$, et $(m, r) = \mathcal{P}^{-1}(p)$.

Ajouter $(m, r, \perp, c)$ à $\mathcal{P}$-List.

Challengeur

Des deux messages (m_0, m_1), choisir un bit b et définir $m^\star = m_b$, puis choisir aléatoirement $r^\star$.

►**Règle** $\mathsf{Chal}^{(5)}$

| $p^\star = \mathcal{P}(m^\star, r^\star)$; $c^\star = \varphi_{\mathsf{pk}}(p^\star)$.

►**Règle** $\mathsf{ChalAdd}^{(5)}$

| Ajouter $(m^\star, r^\star, p^\star, c^\star)$ à $\mathcal{P}$-List.

Répondre $c^\star$

FIG. 5.1 – Simulation formelle dans le jeu $\mathsf{IND-CCA}$ contre φ-FDP

5.2.3 Idée de la preuve

L'objectif est de simuler les oracles $\mathcal{P}$, $\mathcal{P}^{-1}$ et $\mathcal{D}_{\mathsf{sk}}$ de telle manière que l'attaquant ne puisse distinguer les simulations des oracles réels. Dans cette simulation, la réponse de l'algorithme de déchiffrement à un texte chiffré, qui n'avait pas été obtenu, se veut être une nouvelle valeur aléatoire (indépendante des autres). Afin d'obtenir cette propriété, nous devons rendre la simulation de la permutation aléatoire cohérente. D'autre part, le challenge sera rendu indépendant des textes clairs m_0 et m_1 : l'attaquant n'aura aucun avantage.

La preuve est constituée des modifications successives des règles appliquées dans la simulation (parfaite), où les oracles $\mathcal{P}$ et $\mathcal{P}^{-1}$ sont initialement simulés par l'utilisation d'une permutation parfaitement aléatoire P et son inverse P^{-1}. Le dernier jeu fournira

une simulation de $\mathcal{D}_{\mathsf{sk}}$ sans inverser φ_{pk}.

La simulation est parfaite à moins que l'attaquant ne fasse une requête sur la préimage du challenge à la permutation aléatoire $\mathcal{P}^{-1}$. Mais dans ce cas, l'attaquant aide aussi à inverser la fonction φ.

La preuve est en fait un cas particulier de celle du théorème 47 qui va être présentée en détails dans le chapitre suivant.

5.3 OAEP 3 tours : un padding générique et efficace

5.3.1 Relâchement de la sécurité CCA

À Eurocrypt '02, An *et al* [2] ont proposé la notion de sécurité « CCA généralisée », où l'attaquant n'a pas le droit de demander des textes chiffrés ayant une certaine *relation* avec le challenge à l'oracle de déchiffrement. Cette relation doit être une relation publique et efficacement calculable. De plus, elle doit respecter l'équivalence de déchiffrement (si deux textes chiffrés sont en relation, ils chiffrent nécessairement le même texte clair). Ce relâchement a pour but de montrer que des bits supplémentaires, qui peuvent être facilement ajoutés ou supprimés, ne rendent pas le schéma théoriquement fragile. En effet, d'un point de vue pratique, ils ne jouent aucun rôle.

Un autre relâchement a été récemment proposé par Canetti *et al* [31]. Il s'agit d'une extension de la relation ci-dessus. Lorsqu'une requête c est soumise à l'oracle de déchiffrement, elle est, dans un premier temps, déchiffrée en m : si m est identique à un des deux textes clairs retournés par l'attaquant à la fin de la première étape (m_0 ou m_1), l'oracle refuse de répondre (retourne $\perp$ ou test), sinon il retourne m. Les auteurs appellent cette variante la sécurité « CCA rejouable ». D'après eux, ce niveau de sécurité, bien que plus faible que le niveau usuel CCA, est suffisant dans presque toutes les applications pratiques.

Dans notre travail, nous pouvions utiliser le deuxième relâchement, le « CCA rejouable ». Cependant, afin d'obtenir une preuve de sécurité plus simple et un résultat plus précis (avec un corollaire intéressant pour les cas particuliers tels que RSA), nous proposons une notion de sécurité plus forte que « CCA rejouable » : « CCA relâchée », dénotée RCCA. Un schéma sûr dans ce scénario l'est dans celui de « CCA rejouable », mais ne l'est pas nécessairement dans celui de « CCA généralisée » ou « CCA classique ». Les relations entre ces scénarios dépendent de la manière dont la chaîne aléatoire est utilisée. Dans la notation formelle de l'algorithme de chiffrement, on découpe en effet la chaîne aléatoire en deux parties r et ρ : $c = \mathcal{E}_{\mathsf{pk}}(m; r, \rho)$. L'algorithme de chiffrement est donc une fonction de $\mathcal{M} \times \mathcal{R} \times \mathrm{R}$ dans l'ensemble des textes chiffrés. Pour garantir la consistance du schéma de chiffrement, elle doit être une injection par rapport à $\mathcal{M}$ (plusieurs éléments de $\mathcal{M} \times \mathcal{R} \times \mathrm{R}$ peuvent correspondre à un même texte chiffré, mais leur projection sur $\mathcal{M}$ doit obligatoirement donner un texte clair unique). Dans le cadre de notre relâchement, la chaîne aléatoire $\mathcal{R} \times \mathrm{R}$ est découpée de telle manière que le chiffrement soit une injection

par rapport à $\mathcal{M} \times \mathcal{R}$.

Soit le challenge $c^\star = \mathcal{E}_{\mathsf{pk}}(m^\star; r^\star, \rho^\star)$. Considérons le texte chiffré $c = \mathcal{E}_{\mathsf{pk}}(m; r, \rho)$, d'après les commentaires précédents, $(m^\star, r^\star)$ et (m, r) sont uniques et définis respectivement à partir de $c^\star$ et c, tandis que $\rho^\star$ et ρ pourraient ne pas être uniques. A la réception de c, un *oracle de déchiffrement relâché* est défini pour vérifier si $(m^\star, r^\star) = (m, r)$ auquel cas il retourne test (*i.e.* il refuse de répondre), sinon, il retourne m.

Définition 42 (CCA relâché) *Dans le scénario « CCA relâché », un attaquant a accès à l'oracle de déchiffrement relâché mais pas à l'oracle de déchiffrement standard.*

Propriété 43 *La sécurité dans le scénario « CCA relâché » implique la sécurité dans le scénario « CCA rejouable ».*

Preuve. Il s'agit d'ici d'une relation triviale car l'oracle à « CCA rejouable » peut être facilement simulé par l'oracle « relâché » : s'il retourne test, cette valeur sera transmise, sinon, la réponse m sera comparée avec les deux textes clairs $\{m_0, m_1\}$, retournés par l'attaquant à la fin de la première étape. A l'issue de cette comparaison, test sera retournée si $m \in \{m_0, m_1\}$, sinon m sera la réponse. □

Cette propriété montre que notre cadre de travail reste acceptable d'un point de vue pratique. De plus, si R est un ensemble vide et si f est une bijection, RCCA sera identique à CCA.

5.3.2 Primitive de base

Notre objectif est de proposer une variante d'OAEP qui peut être utilisée avec une grande classe de fonctions à sens-unique. Plus précisément, on n'a besoin que d'une famille de fonctions injectives, *probabilistes* à sens-unique à trappe. Une telle famille est dénotée (φ_{pk}) de l'ensemble E_{pk} vers l'ensemble F_{pk}, pour tout indice pk. Nous verrons que plusieurs primitives de chiffrement, où les espaces de textes clairs et de textes chiffrés sont respectivement représentés par E_{pk} et F_{pk}, sont convenables : pour chaque pk (la clef publique), il existe une fonction ψ_{sk} (où sk est la clef secrète) qui retourne la pré-image dans E_{pk}. Une famille de fonctions injectives, *probabilistes* à sens-unique à trappe de E vers F contient des fonctions $f : \mathrm{E} \times \mathrm{R} \to \mathrm{F}$, qui prend un couple (x, ρ) et retourne $y \in \mathrm{F}$. L´élément x, dans E, est considéré comme l'entrée et ρ est une chaîne aléatoire dans R qui rend la fonction probabiliste. L'injectivité implique que pour tout y, il existe au plus un x (mais peut-être plusieurs ρ) tel que $y = f(x, \rho)$. La fonction g, sur l'entrée y, retourne x. Cette fonction est l'inverse de la fonction probabiliste f. Clairement, on a besoin que la fonction f soit efficacement calculable et que son inverse soit difficilement calculable sans la trappe (la propriété à sens-unique à trappe). Cependant, pour prouver la sécurité de notre construction, nous avons besoin de deux propriétés additionnelles :

– la fonction $f : \mathrm{E} \times \mathrm{R} \to \mathrm{F}$ est une bijection ;

– sans la connaissance de la trappe, il est difficile d'inverser f, même si on a accès à un oracle décisionnel $\mathsf{Same}_f(y, y')$ qui vérifie si $g(y) = g(y')$.

La deuxième propriété est en effet la notion du problème de « séparation » (« gap problem » en anglais), qui est défini par la probabilité de succès $\mathsf{Succ}_f^{\mathsf{gap}}(t, q)$: pour tout attaquant $\mathcal{A}$ dont le temps de calcul est borné par t, et le nombre de requêtes à l'oracle décisionnel Same_f est borné par q :

$$\mathsf{Succ}_f^{\mathsf{gap}}(t, q) = \max_{\mathcal{A}}\{x \stackrel{R}{\leftarrow} \mathrm{E}, \rho \stackrel{R}{\leftarrow} \mathrm{R}, y = f(x, \rho) : \mathcal{A}^{\mathsf{Same}_f}(y) = x\}.$$

Pour une famille de fonctions (φ_{pk}), on note $\mathsf{Succ}_{\varphi}^{\mathsf{gap}}(t, q)$ la probabilité de succès $\mathsf{Succ}_{\varphi_{\mathsf{pk}}}^{\mathsf{gap}}(t, q)$ où pk est choisie aléatoirement dans l'espace des clefs publiques.

Considérons ces deux propriétés d'un point de vue pratique :

– Le premier exemple à considérer est évidemment la permutation RSA [107]. Pour une clef publique $\mathsf{pk} = (n, e)$ donnée, les ensembles $\mathrm{E}, \mathrm{F}, \mathrm{R}$ sont définis par : $\mathrm{E} = \mathrm{F} = \mathbb{Z}_n^\star$ et R est l'ensemble vide. Il s'agit clairement d'une fonction injective (et déterministe), qui est, en outre, une bijection. Grâce au déterminisme, l'oracle décisionnel $\mathsf{Same}(y, y')$ vérifie simplement si $y = y'$: le problème de séparation RSA est ainsi le problème classique de RSA.
– L'objectif de notre extension d'OAEP est de l'appliquer au chiffrement bien connu d'ElGamal [45] dans un groupe cyclique $\mathbb{G}$ d'ordre q, généré par g. Étant donnée une clef publique $\mathsf{pk} = y \in \mathbb{G}$, les ensembles $\mathrm{E}, \mathrm{R}, \mathrm{F}$ sont définis par : $\mathrm{E} = \mathbb{G}$, $\mathrm{R} = \mathbb{Z}_q$ et $\mathrm{F} = \mathbb{G} \times \mathbb{G}$. La fonction de chiffrement $\varphi_y(x, \rho) = (g^\rho, x \times y^\rho)$ est à la fois une injection probabiliste de E vers F et une bijection de $\mathrm{E} \times \mathrm{R}$ vers F. À propos de l'oracle décisionnel, il devra vérifier, sur les entrées $(a = g^\rho, b = x \times y^\rho)$ et $(a' = g^{\rho'}, b' = x' \times y^{\rho'})$, si $x = x'$, ce qui est équivalent à décider si $(g, y, a'/a = g^{\rho'-\rho}, b'/b = (x'/x) \times y^{\rho'-\rho})$ est un quadruplet Diffie-Hellman : le problème de séparation est ainsi le problème connu Gap Diffie-Hellman [93, 94].
– Avec des arguments similaires, on peut montrer que le chiffrement de Paillier [95] peut également être utilisé avec notre construction.

5.3.3 Description d'OAEP 3 tours

Notations et paramètres communs

Pour une présentation plus simple, et aussi des analyses de sécurité plus claires, nous nous concentrons sur le cas où $\mathrm{E} = \{0, 1\}^n$ (un ensemble binaire). Une analyse du cas plus général est considérée dans [96]. On remarque que la plupart des fonctions peut être, à faible coût, transformées afin de satisfaire cette contrainte [4]. Cette construction est illustrée sur la figure 5.2 Le chiffrement et le déchiffrement utilisent trois fonctions de hachage : $\mathcal{F}, \mathcal{G}, \mathcal{H}$, modélisées ultérieurement dans les analyses de sécurité par des oracles aléatoires. Les paramètres de sécurité satisfont $n = k + \ell$:

$$\mathcal{F} : \{0, 1\}^k \to \{0, 1\}^\ell \qquad \mathcal{G} : \{0, 1\}^\ell \to \{0, 1\}^k \qquad \mathcal{H} : \{0, 1\}^k \to \{0, 1\}^\ell.$$

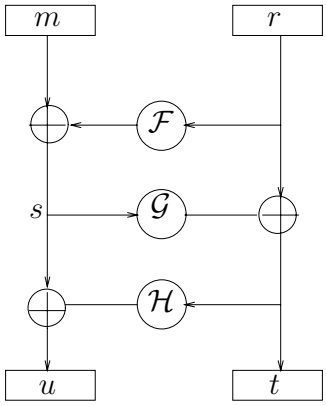

FIG. 5.2 – OAEP 3 tours

Le chiffrement utilise une famille de permutations à sens-unique (φ_{pk}) dont les inverses sont respectivement les ψ_{sk}, où sk est la clef secrète (*i.e.* la trappe) associée à la clef publique pk. Le symbole « $\|$ » dénote la concaténation des chaînes de bits. De plus, on identifie $\{0,1\}^k \times \{0,1\}^\ell$ à $\{0,1\}^n$.

Algorithme de chiffrement

L'espace des textes clairs est $\mathcal{M} = \{0,1\}^\ell$. Le chiffrement utilise deux chaînes aléatoires $r \in \mathcal{R} = \{0,1\}^k$ et $\rho \in \mathrm{R}$. A partir du texte clair $m \in \mathcal{M}$, il retourne un texte chiffré c de F :

$$s = m \oplus \mathcal{F}(r) \qquad t = r \oplus \mathcal{G}(s) \qquad u = s \oplus \mathcal{H}(t) \qquad c = \varphi_{\mathsf{pk}}(u\|t, \rho).$$

Algorithme de déchiffrement

A partir du texte chiffré c, on calcule d'abord $u\|t = \psi_{\mathsf{sk}}(c)$, où $u \in \{0,1\}^\ell$ et $t \in \{0,1\}^k$, puis retrouve le texte clair m par :

$$s = u \oplus \mathcal{H}(t) \qquad r = t \oplus \mathcal{G}(s) \qquad m = s \oplus \mathcal{F}(r).$$

5.4 Résultat de sécurité

Dans cette section, nous présentons le résultat de l'analyse de sécurité du schéma OAEP 3 tours et la preuve de ce résultat.

Théorème 44 *Considérons un attaquant $\mathcal{A}$ contre la construction d'OAEP 3 tours, selon une attaque* IND−RCCA. *Supposons qu'après q_f, q_g, q_h et q_d requêtes respectivement aux*

oracles $\mathcal{F}$, $\mathcal{G}$, $\mathcal{H}$, et à l'oracle de déchiffrement, $\mathcal{A}$ peut avoir un avantage $\mathsf{Adv}^{\mathsf{ind-rcca}}_{\mathsf{oaep\text{-}3}}(\tau)$ *en temps τ, alors* $\mathsf{Succ}^{\mathsf{gap}}_{\varphi}(\tau', q_d(q_g q_h + q_d))$ *est borné par :*

$$\frac{1}{2} \times \mathsf{Adv}^{\mathsf{ind-rcca}}_{\mathsf{oaep\text{-}3}}(\tau') - q_d^2 \times \left(\frac{1}{2^\ell} + \frac{6}{2^k}\right) - (4q_d + 1) \times \left(\frac{q_g}{2^\ell} + \frac{q_f}{2^k}\right) - q_d \times \frac{q_f + 1}{2^k},$$

avec $\tau' \leq \tau + (q_f + q_g + q_h + q_d)T_{lu} + q_d^2 T_{\mathsf{Same}} + (q_d + 1)q_g q_h (T_\varphi + T_{\mathsf{Same}})$, où T_φ est le temps pour une évaluation d'une fonction φ_{pk}, T_{Same} est le temps pour une décision de l'oracle $\mathsf{Same}_{\varphi_{\mathsf{pk}}}$, et T_{lu} est le temps pour la consultation d'un élément dans une liste de taille inférieure au nombre total des requêtes.

5.4.1 Permutations à sens-unique

Avant de prouver ce résultat général, considérons le cas particulier où (φ_{pk}) est une famille de permutations. Le résultat général engendre en effet quelques inconvénients :

- le coût de la réduction présente une facteur cubique $q_d q_g q_h$ qui nécessite l'utilisation de grandes clefs afin d'obtenir un niveau de sécurité convenable.
- le résultat de sécurité repose sur le problème de séparation, qui est une hypothèse forte dans certains cas.
- l'impossibilité d'obtenir le niveau usuel de sécurité $\mathsf{IND{-}CCA}$.

Ces inconvénients sont acceptables car ils sont le prix de la généralisation qui s'applique aux chiffrements d'ElGamal et de Paillier. Cependant, pour les permutations à sens-unique à trappe telles que RSA, plusieurs autres variantes d'OAEP offrent une meilleure efficacité. Mais on devrait interpréter notre résultat dans ce cas particulier des permutations à sens-unique à trappe :

- d'abord, le problème de séparation devient le problème RSA classique à sens-unique grâce au déterminisme de la permutation ;
- le scénario de RCCA devient le scénario CCA classique ;
- finalement, grâce au déterminisme de la permutation, on peut éviter le facteur cubique dans la réduction, et obtenir ainsi le facteur quadratique usuel $q_g q_h$, comme dans toutes les constructions OAEP+, SAEP et SAEP+.

On peut alors obtenir un résultat de sécurité bien meilleur :

Théorème 45 *Considérons un attaquant $\mathcal{A}$ contre la construction d'OAEP 3 tours utilisant une famille de permutations à sens-unique à trappe (φ_{pk}) et selon une attaque* $\mathsf{IND{-}CCA}$. *Supposons qu'après q_D requêtes à l'oracle de déchiffrement et q_f, q_g, q_h requêtes respectivement aux oracles $\mathcal{F}$, $\mathcal{G}$ et $\mathcal{H}$, $\mathcal{A}$ peut avoir un avantage* $\mathsf{Adv}^{\mathsf{ind-cca}}_{\mathsf{oaep\text{-}3}}(\tau)$ *en temps τ, alors on peut inverser φ avec succès* $\mathsf{Succ}^{\mathsf{ow}}_{\varphi}(\tau')$, *en temps τ', borné par :*

$$\frac{1}{2} \times \mathsf{Adv}^{\mathsf{ind-cca}}_{\mathsf{oaep\text{-}3}}(\tau') - q_d^2 \times \left(\frac{1}{2^\ell} + \frac{6}{2^k}\right) - (4q_d + 1) \times \left(\frac{q_g}{2^\ell} + \frac{q_f}{2^k}\right) - q_d \times \frac{q_f + 1}{2^k},$$

avec $\tau' \leq \tau + (q_f + q_g + q_h + q_d)T_{lu} + q_g q_h T_\varphi$, où T_φ est le temps pour une évaluation d'une fonction φ_{pk}, et T_{lu} est le temps pour la consultation d'un élément dans une liste de taille inférieure au nombre total des requêtes.

5.4.2 Idée de la preuve

L'objectif de la preuve est de simuler les oracles de telle manière que l'attaquant ne puisse distinguer les simulations des oracles réels.

La simulation des oracles aléatoires utilise des listes pour stocker les questions-réponses. L'oracle de déchiffrement est simulé comme suit : quand une requête y est faite, si les valeurs s et t correspondantes ont toutes été soumises à $\mathcal{G}$ et à $\mathcal{H}$, on pourra extraire m, sinon on retournera un texte clair aléatoire. Cependant, cette simulation entraîne plusieurs relations implicites entre les oracles aléatoires $\mathcal{F}$, $\mathcal{G}$ et $\mathcal{H}$. Nous prouvons que, si la fonction φ_{pk} est difficile à inverser, les éventuelles contradictions engendrées par ces relations ne peuvent être détectées par l'attaquant.

Il est à noter que nous étudions ici une classe plus générale que les seules permutations : les fonctions probabilistes injectives. Ceci peut provoquer quelques problèmes. En effet, pour une permutation, une pré-image correspond à une image unique. Au contraire, dans le cas d'une fonction f, une pré-image x peut correspondre à plusieurs images $y = f(x, \rho)$, avec ρ une chaîne aléatoire. L'attaquant peut donc construire d'autres c' dont la pré-image est identique à celle du challenge c : $g(c) = g(c')$. Une telle requête c' peut être soumise à l'oracle de déchiffrement dans le scénario CCA, auquel cas on ne peut la détecter et ne peut ainsi donner une réponse cohérente. Pour résoudre ce problème, on a introduit la version relâchée de la sécurité à chiffrés choisis, puis on utilise l'oracle décisionnel Same_f. Cet oracle permet de détecter les textes chiffrés dont la pré-image est identique à celle du challenge et peut donc répondre test, *i.e.* refuser de répondre. Il peut également détecter si une requête à l'oracle de déchiffrement y a la même pré-image qu'une requête précédente de déchiffrement y' : si oui, on retournera le même texte clair ; sinon, y est un nouveau texte chiffré. Dans le cas d'un nouveau texte chiffré, grâce à Same_f, on vérifie si s et t ont été soumises respectivement à $\mathcal{G}$ et à $\mathcal{H}$: si oui, le texte clair m est facilement extrait ; sinon, on retourne un texte clair aléatoire.

5.4.3 Preuve

Nous présentons les principales étapes de la preuve. Elle utilise la technique des jeux successifs pour prouver que le simulateur de déchiffrement, décrit ci-dessus, est calculatoirement indistinguable de l'algorithme de déchiffrement réel pour tout attaquant. Puis, dans ce nouvel environnement, un attaquant $\mathsf{IND{-}RCCA}$ peut être facilement utilisé pour inverser la fonction à sens-unique.

JEU $\mathbf{J}_0$: On considère l'attaquant $A = (A_1, A_2)$ contre ce schéma selon une attaque $\mathsf{IND{-}RCCA}$. Après avoir vu la clef publique (la description de la fonction φ_{pk}), A_1 retourne deux textes clairs (m_0, m_1). Après avoir reçu le chiffré $C^\star = a^\star \,\|\, b^\star \,\|\, c^\star$ du message m_b, A_2 retourne un bit b'. On note $r^\star, t^\star, u^\star$ les uniques éléments tels que $c^\star = \mathcal{E}_{\mathsf{pk}}(m_b, r^\star, \rho^\star) = \varphi_{\mathsf{pk}}(u^\star \| t^\star, \rho^\star)$. Remarquons que l'attaquant a accès à l'oracle de déchiffrement $\mathcal{D}_{\mathsf{sk}}$ et aux oracles aléatoires $\mathcal{F}$, $\mathcal{G}$ et $\mathcal{H}$ à toute étape de l'attaque.

Oracles $\mathcal{F}$, $\mathcal{G}$ et $\mathcal{H}$	Requête $\mathcal{F}(r)$: si r est dans la liste, *i.e.* un couple (r, f) se trouve dans la liste $\mathcal{F}$-List, la réponse sera f. Sinon, la réponse f est aléatoirement choisie dans $\{0,1\}^\ell$ et le couple (r, f) sera ajouté à la $\mathcal{F}$-List. Requête $\mathcal{G}(s)$: si s est dans la liste, *i.e.* un couple (s, g) se trouve dans la liste $\mathcal{G}$-List, la réponse sera g. Sinon, la réponse g est aléatoirement choisie dans $\{0,1\}^k$ et le couple (s, g) sera ajouté à la $\mathcal{G}$-List. ►**Règle** $\mathsf{EvalGAdd}^{(0)}$ Ne rien faire `% Sera défini après` Requête $\mathcal{H}(t)$: si t est dans la liste, *i.e.* un couple (t, h) se trouve dans la liste $\mathcal{H}$-List, la réponse sera h. Sinon, la réponse h est aléatoirement choisie dans $\{0,1\}^\ell$ et le couple (t, h) sera ajouté à la $\mathcal{H}$-List.

FIG. 5.3 – Simulation formelle dans le jeu IND−RCCA : les oracles aléatoires

On note S_0 l'événement $b' = b$ ainsi que S_n dans les jeux $\mathbf{J}_n$ ci-dessous.

$$\Pr[\mathsf{S}_0] = \frac{1}{2} \times \left(\mathsf{Adv}^{\mathsf{ind-rcca}}_{\mathsf{oaep\text{-}3}}(\tau) + 1\right).$$

Avantage nul

L'objectif des deux premiers jeux est de modifier le jeu réel en un jeu dans lequel le bit b est parfaitement indistinguable.

JEU $\mathbf{J}_1$: Les simulations des oracles aléatoires $\mathcal{F}$, $\mathcal{G}$ et $\mathcal{H}$, ainsi que celle de l'oracle de déchiffrement $\mathcal{D}_{\mathsf{sk}}$, sont présentées dans les figures 5.3 et 5.4, et la simulation du challengeur est présentée dans la figure 5.5. Ces simulations sont effectuées grâce à l'utilisation des listes $\mathcal{F}$-List, $\mathcal{G}$-List, $\mathcal{H}$-List et $\mathcal{D}$-List. Leur mission est de répondre de façon cohérente aux requêtes soumises aux oracles aléatoires et à l'oracle de déchiffrement. Il s'agit de simulations parfaites.

JEU $\mathbf{J}_2$: Pour que l'avantage de tout attaquant soit nul, nous définissons le masque $f^\star$ de telle sorte qu'il soit totalement indépendant de la vue de l'attaquant :

►**Règle** $\mathsf{Chal}^{(2)}$

Les deux valeurs $r^+ \xleftarrow{R} \{0,1\}^k$ et $f^+ \xleftarrow{R} \{0,1\}^\ell$ sont choisies de façon aléatoire, puis on définit $r^\star = r^+, f^\star = f^+$ et $s^\star = m^\star \oplus f^+, g^\star = \mathcal{G}(s^\star), t^\star = r^+ \oplus g^\star, h^\star = \mathcal{H}(t^\star), u^\star = s^\star \oplus h^\star$.

Oracle $\mathcal{D}$

Requête $\mathcal{D}_{\mathsf{sk}}(c)$: Cas particulier : pour une requête dans la deuxième étape, une requête $(c, c^\star)$ est faite à l'oracle décisionnel $\mathsf{Same}_{\varphi_{\mathsf{pk}}}$. Si la décision est positive, la réponse sera test et la simulation est terminée. Sinon, on se trouve dans le cas général.
Cas général : pour chaque (m', c') dans $\mathcal{D}$-List, soumettre (c, c') à l'oracle décisionnel $\mathsf{Same}_{\varphi_{\mathsf{pk}}}$. Dans le cas d'une décision positive, la réponse sera m'.
Dans le cas contraire, la réponse est m et est définie par les règles suivantes :

►**Règle** Decrypt−Init$^{(0)}$

Calculer $u \| t = \psi_{\mathsf{sk}}(c)$;

– Si t est dans la liste, *i.e.* un couple (t, h) se trouve dans $\mathcal{H}$-List, calculer $s = u \oplus h$.

– si s est dans la liste, *i.e.* un couple (s, g) se trouve dans $\mathcal{G}$-List, calculer $r = t \oplus g$.

– si r est dans la liste, *i.e.* un couple (r, f) se trouve dans $\mathcal{F}$-List

►**Règle** Decrypt−TSR$^{(0)}$

$h = \mathcal{H}(t)$,
$s = u \oplus h, \quad g = \mathcal{G}(s)$,
$r = t \oplus g, \quad f = \mathcal{F}(r)$,
$m = s \oplus f$.

– sinon

►**Règle** Decrypt−TSnoR$^{(0)}$

même règle que Decrypt−TSR$^{(0)}$.

– sinon

►**Règle** Decrypt−TnoS$^{(0)}$

même règle que Decrypt−TSR$^{(0)}$.

– sinon

►**Règle** Decrypt−noT$^{(0)}$

même règle que Decrypt−TSR$^{(0)}$.

Répondre m et ajouter (m, c) à la $\mathcal{D}$-List.

FIG. 5.4 – Simulation formelle dans le jeu IND−RCCA : l'oracle de déchiffrement

Challengeur

Pour deux messages (m_0, m_1), lancer une pièce à pile ou face pour b et poser $m^\star = m_b$, choisir aléatoirement $r^\star$ puis répondre $c^\star$ où :

▶**Règle** $\mathsf{Chal}^{(0)}$

$f^\star = \mathcal{F}(r^\star), \quad s^\star = m^\star \oplus f^\star,$
$g^\star = \mathcal{G}(s^\star), \quad t^\star = r^\star \oplus g^\star,$
$h^\star = \mathcal{H}(t^\star), \quad u^\star = s^\star \oplus h^\star.$

▶**Règle** $\mathsf{ChalC}^{(0)}$

et $c^\star = \varphi_{\mathsf{pk}}(u^\star \| t^\star, \rho^\star)$, pour une chaîne aléatoire $\rho^\star$.

FIG. 5.5 – Simulation formelle dans le jeu IND−RCCA : le générateur du challenge

Les deux jeux $\mathbf{J}_2$ et $\mathbf{J}_1$ sont parfaitement indistinguables sauf si $r^\star$ a été soumise à $\mathcal{F}$ (par l'attaquant ou par l'oracle de déchiffrement). On note AskF_2 cet événement ainsi que AskF_n dans les jeux $\mathbf{J}_n$ ci-dessous.

$$|\Pr[\mathsf{S}_2] - \Pr[\mathsf{S}_1]| \leq \Pr[\mathsf{AskF}_2].$$

Comme souhaité, f^+ est utilisée dans la génération du challenge pour masquer le texte clair, mais n'apparaît pas ailleurs puisque $\mathcal{F}(r^+)$ n'est plus défini comme étant f^+. Ainsi, la réponse de $\mathcal{A}_2$ suit une distribution indépendante de b : $\Pr[\mathsf{S}_2] = 1/2$. On obtient la première conclusion :

$$\mathsf{Adv}^{\mathsf{ind-rcca}}_{\mathsf{oaep\text{-}3}}(t) \leq 2 \times \Pr[\mathsf{AskF}_2]. \tag{5.1}$$

L'objectif est désormais d'étudier l'événement AskF.

Exclusion des requêtes aux oracles $\mathcal{G}$ et $\mathcal{H}$ dans la simulation du déchiffrement

JEU $\mathbf{J}_3$: On commence la phase de simulation de l'oracle de déchiffrement. Tout d'abord, les règles Decrypt−noT, Decrypt−TnoS et Decrypt−TSnoR sont modifiées de telle manière qu'elles retournent des messages aléatoires : les réponses des oracles $\mathcal{F}$, $\mathcal{G}$ et $\mathcal{H}$ sont remplacées par des valeurs aléatoires.

▶**Règle** $\mathsf{Decrypt-noT}^{(3)}$

Choisir $m \xleftarrow{R} \{0,1\}^\ell$, $h \xleftarrow{R} \{0,1\}^\ell$ et $g \xleftarrow{R} \{0,1\}^k$
Définir $s = u \oplus h$, $r = t \oplus g$ et calculer $f = m \oplus s$.
Ajouter (r, f) à la $\mathcal{F}$-List, (s, g) à la $\mathcal{G}$-List, (t, h) à la $\mathcal{H}$-List.

▶**Règle** $\mathsf{Decrypt-TnoS}^{(3)}$

Choisir $m \xleftarrow{R} \{0,1\}^\ell$ et $g \xleftarrow{R} \{0,1\}^k$
Définir $r = t \oplus g$ et calculer $f = m \oplus s$.
Ajouter (r, f) à la $\mathcal{F}$-List et (s, g) à la $\mathcal{G}$-List.

►**Règle** Decrypt−TSnoR$^{(3)}$

Choisir $m \xleftarrow{R} \{0,1\}^\ell$.
Calculer $f = m \oplus s$.
Ajouter (r, f) à la $\mathcal{F}$-List.

Ces règles sont en général similaires à celles utilisées dans les jeux précédents. La relation suivante peut être facilement démontrée (les détails sont dans l'article [98]) :

$$|\Pr[\mathsf{AskF}_3] - \Pr[\mathsf{AskF}_2]| \leq q_d \times \left(\frac{q_g + q_d}{2^\ell} + 2 \times \frac{q_f + q_d}{2^k}\right). \tag{5.2}$$

JEU $\mathbf{J}_4$: Dans les jeux précédents, les simulations des oracles étaient quasi-parfaites grâce à l'ajout de nouvelles relations aux listes correspondantes. Dans ce jeu, on effectue des modifications plus techniques : $\mathcal{G}$-List ne stocke plus les nouvelles relations (s, g). Par conséquent, g n'est plus explicitement définie, la valeur de r correspondante ne peut plus être calculée, et (r, f) ne sera plus dans la $\mathcal{F}$-List. Cependant, dès que $\mathcal{G}(s)$ est connue, on peut définir $\mathcal{F}(r)$ de façon cohérente et mettre à jour les listes :

►**Règle** Decrypt−TnoS$^{(4)}$

Choisir $m \xleftarrow{R} \{0,1\}^\ell$.

►**Règle** Decrypt−noT$^{(4)}$

Choisir $h \xleftarrow{R} \{0,1\}^\ell$ et $m \xleftarrow{R} \{0,1\}^\ell$.
Ajouter (t, h) à la $\mathcal{H}$-List.

►**Règle** EvalGAdd$^{(4)}$

Pour chaque $(t, h) \in \mathcal{H}$-List et chaque $(m, c) \in \mathcal{D}$-List, choisir une chaîne aléatoire $\rho \in \mathrm{R}$ et demander $(c, c' = \varphi_{\mathsf{pk}}(h \oplus s \| t, \rho))$ à l'oracle décisionnel $\mathsf{Same}_{\varphi_{\mathsf{pk}}}$. Si $\mathsf{Same}_{\varphi_{\mathsf{pk}}}$ retourne une réponse positive, on calcule $r = t \oplus g$ et $f = m \oplus s$ et on ajoute (r, f) à la $\mathcal{F}$-List.

Avec ces nouvelles règles, les réponses données dans $\mathbf{J}_3$ et $\mathbf{J}_4$ sont parfaitement indistinguables, à l'exception du cas où r est soumise à $\mathcal{F}$ avant la soumission de s à $\mathcal{G}$. On note cet événement AskRbS_4 ainsi que AskRbS_n dans les jeux $\mathbf{J}_n$ ci-dessous. Quand r est soumise après s, la simulation est parfaite : au moment où s est soumise, grâce à la simulation de $\mathcal{G}$ (et la règle supplémentaire EvalGAdd) on trouve (t, h) et ainsi (r, f) est calculé d'une manière cohérente, exactement comme dans $\mathbf{J}_3$, et ajouté à $\mathcal{F}$-List.

Remarquons que pour chaque texte chiffré c, t est unique, par conséquent, h et s le sont aussi. Alors, le cas d'échec apparaît au plus une fois pour chaque texte chiffré soumis à l'oracle de déchiffrement.

Cependant, tant que s n'est pas soumise, g est une variable aléatoire uniformément distribuée. Par conséquent, r l'est aussi. La probabilité d'échec (*i.e.* r a déjà été soumise à $\mathcal{F}$) est ainsi $q_f/2^k$:

$$\Pr[\mathsf{AskRbS}_4] \leq q_d \times \frac{q_f + q_d}{2^k}.$$

Un problème survient en raison de la suppression de quelques éléments (s, g) de $\mathcal{G}$-List, et (r, f) de $\mathcal{F}$-List. Cette suppression peut éventuellement avoir un impact sur la simulation des requêtes de déchiffrement et sur l'événement AskF :

- si le couple (r, f) où $r = r^\star$ est supprimé, l'événement AskF ne peut se produire que dans $\mathbf{J}_3$ mais pas dans $\mathbf{J}_4$. Heureusement, puisque $r = t \oplus g$, où g est aléatoirement choisie, la probabilité d'occurrence de cet événement est au plus $1/2^k$.
- quand on simule une requête de déchiffrement ultérieure $c' = \varphi_{\mathsf{pk}}(u' \| t', \rho')$, l'élément $s' = s$ peut être dans $\mathcal{G}$-List de $\mathbf{J}_3$ mais pas dans celle de $\mathbf{J}_4$. Par conséquent, la règle Decrypt−TnoS est utilisée au lieu de Decrypt−TSR ou Decrypt−TSnoR. g est donc définie lors du premier déchiffrement dans $\mathbf{J}_3$ mais ne serait jamais révélée. La probabilité que $r' = t' \oplus g' = t' \oplus g$ ne soit pas dans la liste $\mathcal{F}$-List est donc $(q_f + q_d)/2^k$ (modification de Decrypt−TSR à Decrypt−TnoS). Dans le cas où r' n'est pas dans $\mathcal{F}$-List, m' est aléatoire à la fois dans $\mathbf{J}_3$ et dans $\mathbf{J}_4$: un passage de Decrypt−TSnoR à Decrypt−TnoS.

$$|\Pr[\mathsf{AskF}_4] - \Pr[\mathsf{AskF}_3]| \leq \Pr[\mathsf{AskRbS}_4] + q_d \times \frac{q_f + q_d}{2^k} + q_d \times \frac{1}{2^k} \leq 2q_d \times \frac{q_f + q_d}{2^k} + q_d \times \frac{1}{2^k}.$$

Jeu $\mathbf{J}_5$: On continue à simplifier la simulation de l'oracle de déchiffrement : les nouvelles relations (t, h) ne sont plus stockées dans $\mathcal{H}$-List.

▶**Règle** Decrypt−noT$^{(5)}$

Choisir $m \xleftarrow{R} \{0,1\}^\ell$.

Par rapport à $\mathbf{J}_4$, les réponses des simulations de l'oracle de déchiffrement dans $\mathbf{J}_5$ sont identiques (valeurs aléatoires). Néanmoins, $\mathcal{H}$-List a changé et ce changement peut avoir quelques impacts :

- $\mathcal{F}$ peut répondre différemment. On note AskSbT_5, ainsi que AskSbT_n dans les jeux $\mathbf{J}_n$ ci-dessous, l'événement où s est soumise à $\mathcal{G}$ avant que t ne soit soumise à $\mathcal{H}$. Quand AskSbT_5 se produit, la règle EvalGAdd n'est plus applicable. Heureusement, tant que t n'est pas soumise à $\mathcal{H}$, h est une variable aléatoire uniformément distribuée, et donc $s = u \oplus h$ l'est également. Par conséquent, la probabilité d'occurrence de AskSbT_5 est égale à $q_g/2^\ell$ (puisqu'aucune nouvelle relation de $\mathcal{G}$ n'est ajoutée par la simulation de déchiffrement) :

$$\Pr[\mathsf{AskSbT}_5] \leq q_d \times \frac{q_g}{2^\ell}.$$

- la suppression de l'élément (t, h) de la $\mathcal{H}$-List peut éventuellement avoir un impact sur la simulation d'une requête de déchiffrement ultérieure $c' = \varphi_{\mathsf{pk}}(u' \| t', \rho')$:
  - si s' est dans $\mathcal{G}$-List et $t' = t$ a été trouvée dans le jeu $\mathbf{J}_4$. Comme t' n'est plus dans $\mathcal{H}$-List, la règle Decrypt−noT est utilisée au lieu de Decrypt−TSR ou Decrypt−TSnoR. Dans ce cas, $h' = h$ est définie lors du premier déchiffrement

dans $\mathbf{J}_4$ mais elle n'est plus révélée. La probabilité que $s' = t' \oplus h$ soit dans $\mathcal{G}$-List est au plus $q_g/2^\ell$.

– si s' n'est pas dans $\mathcal{G}$-List mais $t' = t$ a été trouvée dans le $\mathbf{J}_4$. Comme t' n'est plus dans $\mathcal{H}$-List, la règle Decrypt−noT est utilisée au lieu de Decrypt−TnoS. Dans ce cas, le déchiffrement reste le même car comme dans le jeu $\mathbf{J}_4$, il retourne un texte clair aléatoire et n'ajoute rien aux listes.

On obtient finalement :

$$|\Pr[\mathsf{AskF}_5] - \Pr[\mathsf{AskF}_4]| \leq \Pr[\mathsf{AskSbT}_5] + q_d \times \frac{q_g}{2^\ell} \leq 2q_d \times \frac{q_g}{2^\ell}.$$

Remarquons que les listes $\mathcal{G}$-List et $\mathcal{H}$-List ne contiennent maintenant que les requêtes posées par l'attaquant et par la génération du challenge. La simulation de déchiffrement ne fait plus appel aux oracles $\mathcal{G}$ ou $\mathcal{H}$, mais seulement à $\mathcal{F}$.

On note AskGA_5 (resp. AskHA_5) l'événement où $s^\star$ (resp. $t^\star$) (valeur obtenue lors de la génération du challenge), est soumise par l'attaquant. On désigne aussi AskGHA_5 l'événement où les deux événements AskGA_5 et AskHA_5 se produisent.

Extracteur du texte clair

On complète maintenant la simulation de l'oracle de déchiffrement en rendant son comportement identique à celui d'un extracteur classique de textes clairs, brièvement décrit dans l'« idée de la preuve ».

Jeu $\mathbf{J}_6$: Avant de faire des modifications, on introduit la règle Abort pour supprimer quelques exécutions. Si une des conditions d'application de cette règle est vérifiée, le jeu s'arrête et on retourne une réponse aléatoire b'.

►**Règle** $\mathsf{Abort}^{(6)}$

Si $\mathsf{AskGA}_6 \wedge \neg\mathsf{AskHA}_6$.

Si $\neg\mathsf{AskHA}_6$, $\mathcal{H}(t^\star) = u^\star \oplus m_b \oplus f^+$ ne sera jamais révélée. Puisque f^+ est une valeur aléatoire indépendante de la vue de l'attaquant, $\mathcal{H}(t^\star)$ est donc une variable aléatoire uniformément distribuée et $s^\star = u^\star \oplus H(t^\star)$ l'est également. Ceci implique que $s^\star$ ne peut être soumise qu'avec une probabilité $q_g/2^\ell$.

$$|\Pr[\mathsf{AskF}_6] - \Pr[\mathsf{AskF}_5]| \leq \frac{q_g}{2^\ell}.$$

De plus, $\Pr[\mathsf{AskF}_6]$ peut être facilement bornée par la relation suivante dont la preuve se trouve dans l'article [98] :

$$\Pr[\mathsf{AskF}_6] \leq \frac{q_f}{2^k} + \Pr[\mathsf{AskGHA}_6]. \tag{5.3}$$

Conclusion intermédiaire :

$$\Pr[\mathsf{AskF}_2] \leq q_d^2 \times \left(\frac{4}{2^k} + \frac{1}{2^\ell}\right) + (3q_d + 1) \times \left(\frac{q_f}{2^k} + \frac{q_g}{2^\ell}\right) + q_d \times \frac{q_f + 1}{2^k} + \Pr[\mathsf{AskGHA}_6]. \tag{5.4}$$

On s'intéresse désormais à l'événement AskGHA_6.

JEU $\mathbf{J}_7$: Quelques autres exécutions supplémentaires seront également supprimées si une des conditions suivantes est vérifiée :

▶**Règle** $\mathsf{Abort}^{(7)}$

- Si $\mathsf{AskGA}_7 \wedge \neg\mathsf{AskHA}_7$.
- Si la règle $\mathsf{Decrypt{-}TSR/Decrypt{-}TSnoR}$ est appliquée où $t = t^\star$, et $\mathcal{H}(t^\star)$ n'a pas encore été soumise par l'attaquant.
- Si la règle $\mathsf{Decrypt{-}TSR}$ est appliquée où $s = s^\star$, et que $\mathcal{G}(s^\star)$ n'a pas encore été soumise par l'attaquant.

La différence suivante entre ces deux jeux est expliquée dans l'article [98] :

$$|\Pr[\mathsf{AskGHA}_7] - \Pr[\mathsf{AskGHA}_6]| \leq q_d \times \left(\frac{q_f + q_d}{2^k} + \frac{q_g}{2^\ell}\right). \tag{5.5}$$

JEU $\mathbf{J}_8$: On complète la simulation de l'oracle de déchiffrement en la rendant indépendante des requêtes faites par la génération du challenge. Si l'attaquant ne demande pas de requête sur $s^\star$, l'oracle de déchiffrement n'utilise plus $(s^\star, g^\star)$.

▶**Règle** $\mathsf{Decrypt{-}TSnoR}^{(8)}$

Si $s = s^\star$ et que $s^\star$ n'a pas été directement soumise par l'attaquant : $m \xleftarrow{R} \{0,1\}^\ell$.
Sinon, on choisit $m \xleftarrow{R} \{0,1\}^\ell$ et calcule $f = m \oplus s$ et on ajoute (r, f) à la $\mathcal{F}$-List.

Si $s = s^\star$ mais $s^\star$ n'a pas été directement soumise par l'attaquant, en application de la règle $\mathsf{Decrypt{-}TSnoR}$ dans $\mathbf{J}_7$, $f = \mathcal{F}(r)$ est une variable aléatoire uniformément distribuée. On peut donc retourner une réponse aléatoire m. Cependant, dans ce cas, (r, f) ne sera plus stocké et ceci peut causer quelques problèmes si la valeur r' d'une requête de déchiffrement ultérieure c' est identique à r. Comme cela signifie que $g^\star \oplus t = g' \oplus t'$:

- si $s' = s$, alors $t' = t$, ceci implique que la réponse m' est identique à m. Cette éventualité peut cependant être détectée dès le début de la simulation grâce à l'utilisation de l'oracle décisionnel $\mathsf{Same}_{\varphi_{\mathsf{pk}}}$ sur (c, c').
- si $s' \neq s$, on obtient que $g = \mathcal{G}(s) = g^\star$ est indépendante de g'. De plus, $s^\star$ n'est pas soumise, donc $r = g^\star \oplus t$ est une variable aléatoire uniformément distribuée : la probabilité d'occurrence d'une telle requête de déchiffrement c' est $1/2^k$.

On a :

$$|\Pr[\mathsf{AskGHA}_8] - \Pr[\mathsf{AskGHA}_7]| \leq \frac{q_d^2}{2^k}.$$

JEU $\mathbf{J}_9$: Dans $\mathbf{J}_8$, la simulation de l'oracle de déchiffrement n'utilise plus les requêtes faites aux oracles $\mathcal{G}$ et $\mathcal{H}$ par le générateur du challenge :

- $\mathsf{Decrypt{-}TSR}$
  - $r = r^\star$: impossible si la requête $r^\star$ n'a pas encore été soumise directement par l'attaquant, car elle n'a pas été soumise au cours de la génération du challenge ;

  - $s = s^\star$: exclu dans le jeu $\mathbf{J}_7$;
  - $t = t^\star$: exclu dans le jeu $\mathbf{J}_7$;
- Decrypt−TSnoR
  - $s = s^\star$: similaire à Decrypt−TnoS du jeu $\mathbf{J}_8$;
  - $t = t^\star$: exclu dans le jeu $\mathbf{J}_7$;
- Decrypt−TnoS
  - $t = t^\star$: similaire à Decrypt−noT du jeu $\mathbf{J}_5$;

On peut donc modifier la simulation du challenge sans demander de requêtes à $\mathcal{G}$ ou à $\mathcal{H}$:

►**Règle** Chal$^{(9)}$

> Les deux valeurs $r^+ \xleftarrow{R} \{0,1\}^k$ et $f^+ \xleftarrow{R} \{0,1\}^\ell$ sont données, ainsi que $g^+ \xleftarrow{R} \{0,1\}^k$ et $h^+ \xleftarrow{R} \{0,1\}^\ell$ et puis on définit $r^\star = r^+$, $f^\star = f^+$, $s^\star = m^\star \oplus f^+$, $g^\star = g^+$, $t^\star = r^+ \oplus g^\star$, $h^\star = h^+$ et $u^\star = s^\star \oplus h^\star$.

Cette règle n'a aucun impact ni sur la simulation de déchiffrement ni sur la règle EvalGAdd car la modification a déjà été prise en compte dans $\mathbf{J}_5$. Ainsi, les distributions de probabilité ne sont pas modifiées.

Dans ce jeu, la simulation de l'oracle déchiffrement sur c est en effet l'extracteur simple de textes clairs [10, 54, 96]. En utilisant l'oracle décisionnel $\mathsf{Same}_{\varphi_{\mathsf{pk}}}$, elle obtient les valeurs (s, g) et (t, h) correspondant à $c = \varphi_{\mathsf{pk}}(s \oplus h \| t, \rho)$ grâce à une consultation de $\mathcal{G}$-List et $\mathcal{H}$-List (qui ne contiennent maintenant que des requêtes directement faites par l'attaquant). Il est à noter qu'elle n'a pas besoin de ψ_{sk} pour cela :

►**Règle** Decrypt−Init$^{(9)}$

> Pour chaque $(s, g) \in \mathcal{G}$-List et $(t, h) \in \mathcal{H}$-List : on choisit arbitrairement $\rho \in \mathrm{R}$, calcule $c' = \varphi_{\mathsf{pk}}(s \oplus h \| t, \rho)$ et on demande (c, c') à l'oracle décisionnel $\mathsf{Same}_{\varphi_{\mathsf{pk}}}$.
> - s'il existe (s, g) et (t, h) tels que $\mathsf{Same}_{\varphi_{\mathsf{pk}}}$ retourne une réponse affirmative, on définit $u = s \oplus h$.
> - sinon, on pose $t = \perp$ et $u = \perp$.

Le fait de définir $t = \perp$ et $u = \perp$ a pour but de rendre la réponse m aléatoire. Le temps de calcul est donc borné par $q_g q_h \times (T_\varphi + T_{\mathsf{Same}})$ plus le temps de consultation initiale de $\mathcal{D}$-List : $T_{lu} + q_d T_{\mathsf{Same}}$, où T_φ est le temps pour évaluer une fonction dans la famille φ, et T_{Same} est le temps d'exécution de l'oracle décisionnel. Alors, le temps total d'exécution (y compris toutes les consultation dans les listes) est borné par :

$$\tau' \leq \tau + q_d q_g q_h \times (T_\varphi + T_{\mathsf{Same}}) + q_d^2 \times T_{\mathsf{Same}} + (q_f + q_g + q_h + q_d) \times T_{lu}.$$

5.4.4 Cas particulier

Dans le cas où (φ_{pk}) est une famille de permutations, ce résultat peut être amélioré grâce à l'utilisation d'une liste supplémentaire de taille $q_g q_h$, qui stocke des multiplets $(s, g = \mathcal{G}(s), t, h = \mathcal{H}(t), c' = \varphi_{\mathsf{pk}}(s \oplus h \| t))$. Le temps de calcul est réduit à $\tau + q_g q_h \times T_\varphi + (q_f + q_g + q_h + q_d) \times T_{lu}$. De plus, le scénario RCCA devient le scénario classique CCA. On obtient donc un meilleur résultat en terme de sécurité (voir le théorème 45).

5.5 Conclusion

Toute variante d'OAEP [114, 19] appliquée à RSA avec des exposants généraux (*i.e.*, ni Rabin ni $e = 3$) admet, dans le cas optimal, une réduction quadratique en temps au problème RSA [101]. OAEP est même moins efficace à cause de sa réduction au problème d'inverser une permutation à sens-unique partiel. De plus, pour un niveau de sécurité de 2^{-k}, une chaîne aléatoire de taille $2k$ bits ainsi qu'une redondance de taille k bits sont exigées.

Dans ce chapitre, nous avons montré qu'OAEP 3 tours admet une réduction aussi efficace que celle de la meilleure des variantes d'OAEP. En particulier, notre construction n'exige pas de redondance : on gagne donc k bits. Cependant, ceci n'est pas l'avantage principal.

Selon tous les critères, OAEP 3 tours est au moins aussi efficace que les autres variantes d'OAEP. De plus, en pratique :

- grâce à l'absence de redondance, la mise en œuvre est facilitée, en particulier, pour le processus de déchiffrement [79] ;
- des familles de fonctions probabilistes à sens-unique peuvent être utilisées. Cela rend ce schéma plus flexible : il peut être utilisé avec différentes primitives comme les chiffrements d'El-Gamal et de Paillier.

En guise de conclusion, OAEP 3 tours est le padding le plus générique et le plus simple à pouvoir être utilisé avec plusieurs primitives de chiffrement asymétrique.

6

Padding optimal pour le chiffrement et la signature

Sommaire

La sécurité forte va souvent de pair avec d'importantes contraintes concernant la construction des schémas cryptographiques : la sécurité sémantique implique que le chiffrement soit probabiliste et la sécurité contre les falsifications existentielles implique que les schémas de signature aient des redondances.

Il y a quelques années, Coron *et. al* [36] ont suggéré une construction commune pour le chiffrement et la signature : le « *padding universel* ». Cependant, comme le padding universel doit contenir à la fois un aléa et une redondance, le chiffrement et la signature qui en résulte ne sont pas optimaux.

Dans ce chapitre, nous raffinons cette notion de padding universel en permettant à une des parties d'être une chaîne aléatoire ou une chaîne de zéros, *i.e.* une certaine redondance. Ceci nous aide à construire, à partir d'un padding unique, un chiffrement et une signature efficaces : dans un premier temps, dans le modèle de la permutation aléatoire, et dans un deuxième temps, dans celui de l'oracle aléatoire. Dans les deux cas, nous étudions la taille effective des paramètres pour un niveau de sécurité spécifique et nous montrerons par la suite que le premier schéma est optimal en taille.

6.1 Introduction

Pour le chiffrement à clef publique, la notion de base reste la sécurité sémantique selon des attaques à chiffrés choisis [104]. De la même façon, pour la signature, l'exigence est désormais la sécurité contre les falsifications existentielles face aux attaques à messages choisis [61]. Cependant, la sécurité forte ne suffit pas, elle doit, de surcroît, être obtenue de manière efficace selon plusieurs critères : le temps de calcul, la taille du chiffré/de la signature, et la taille du code.

Les deux premiers critères cités sont courants. En effet, il existe dans la littérature des paddings rapides pour le chiffrement [10, 94] et pour la signature [11]. Concernant la bande passante, dans le chapitre précédent, nous avons proposé un padding optimal qui évite la redondance dans le chiffrement. De nombreux schémas de signature avec « reconstitution de message » (*message recovery* en anglais) [92, 86, 11] peuvent améliorer la bande de passante sans pour autant atteindre les solutions optimales en raison de la présence d'aléa et de redondance. Une exception à remarquer, fruit d'une idée récente de Katz et Wang, atteint une bonne réduction de sécurité en utilisant la construction FDH (« Full-Domain Hash » [11]) avec un seul bit additionnel dépendant du texte clair [71].

Pour le troisième critère, *i.e.* la taille du code, Coron *et. al* [36] ont introduit la notion de padding universel : la taille du code est réduite grâce à l'utilisation d'un padding commun pour le chiffrement et la signature. Ils ont proposé une variante de PSS, appelée PSS-ES pour la construction du padding universel. D'autres solutions, dont celle de Komano et Ohta [76], ont ensuite été proposées. Cependant, dans toutes ces constructions, le chiffrement contient de la redondance et la signature est probabiliste.

6.1.1 Redondance et aléa

Comme présenté dans le chapitre précédent, afin d'atteindre la sécurité sémantique, un chiffrement doit être probabiliste. De plus, les redondances sont souvent ajoutées aux chiffrés pour une preuve de connaissance du texte clair ou « plaintext awareness » [10, 7, 37]. Nous avons également prouvé que la redondance pouvait être évitée, au moins dans le modèle de l'oracle aléatoire et dans celui du chiffrement idéal.

De la même façon, afin d'obtenir la sécurité contre les falsifications, une certaine

redondance doit être introduite dans le couple (message, signature) ou dans une chaîne unique en cas de reconstitution de message : sans la clef de signature, il est difficile de créer un couple (message, signature) qui satisfasse la redondance. La plupart des schémas de signature sont, de plus, probabilistes [111, 89, 11, 38] bien que ceci ne soit pas une condition nécessaire (la signature FDH est déterministe, cependant, la réduction n'est pas efficace).

6.1.2 Padding universel

Il convient de noter que le padding universel est applicable en même temps au chiffrement et à la signature, avec les mêmes clefs d'utilisateur : la clef publique est utilisée pour le chiffrement et pour la vérification ; la clef privée, pour le déchiffrement et pour la signature. Même si ceci n'est pas obligatoire, dans la suite, nous considérons ce cas qui est le pire.

En effet, dans le modèle de sécurité, nous considérons les attaquants (contre la sécurité sémantique ou la falsification existentielle) ayant accès au déchiffrement et à la signature. L'oracle de déchiffrement pourra donc les aider à forger des signatures, puisque la même clef est utilisée, et réciproquement au sujet de la sécurité du chiffrement.

6.2 Modèle de sécurité pour la signature

Les schémas de signature sont les versions électroniques des signatures manuscrites pour les documents numériques : la signature d'un utilisateur sur un message m est une chaîne qui dépend de m et de ses données publiques et secrètes. N'importe qui peut vérifier la validité de la signature en n'utilisant que des données publiques.

Nous rappelons brièvement les principales notions de sécurité pour la signature [61].

Un schéma de signature $\mathsf{S} = (\mathcal{K}, \mathcal{S}, \mathcal{V})$ est défini par trois algorithmes :
- L'*algorithme de génération de clef* $\mathcal{K}$: sur l'entrée 1^k, où k est le paramètre de sécurité, l'algorithme probabiliste $\mathcal{K}$ génère un couple clef publique, clef secrète $(\mathsf{pk}, \mathsf{sk})$. On appelle k le paramètre de sécurité. Les tailles des clefs ainsi que les autres paramètres dans le schéma dépendent de ce paramètre.
- L'*algorithme de signature* $\mathcal{S}$: étant donné un message m (dans l'espace des textes clairs $\mathcal{M}$) et un couple de clef publique-clef secrète $(\mathsf{pk}, \mathsf{sk})$, $\mathcal{S}$ produit une signature σ. L'algorithme de signature peut être probabiliste.
- L'*algorithme de vérification* $\mathcal{V}$: étant donné une signature σ, un (ou une partie, éventuellement vide du) message m, et une clef publique pk, $\mathcal{V}$ vérifie si σ est une signature valide et finalement, extrait le message m. En règle générale, l'algorithme de vérification n'a pas à être probabiliste.

6.2.1 Infalsifiabilité

L'objectif de l'attaquant est de produire un nouveau couple (message, signature) valide. Cette production s'appelle la *falsifiabilité existentielle*. La sécurité contre ce type d'attaque est l'*infalsifiabilité existentielle* (dénotée EUF). En ce qui concerne le moyen d'attaque, on considère habituellement les *attaques adaptatives à messages choisis* (dénotée CMA). Dans ce modèle, l'attaquant peut demander au signataire de signer n'importe quel message et adapter ses questions aux réponses obtenues. A la fin, l'attaquant retourne un couple (m, σ), avec m n'ayant jamais été soumis à l'oracle de signature. L'attaquant gagne si σ est une signature valide du message m. L'objectif est de résister aux falsifications existentielles selon des attaques adaptatives à messages choisis (EUF/CMA), *i.e.* la probabilité de succès de tout attaquant $\mathcal{A}$ en un temps raisonnable est négligeable. Cette probabilité de succès est définie comme suit :

$$\mathsf{Succ}_{\mathsf{S}}^{\mathsf{euf/cma}}(\mathcal{A}) = \Pr\left[\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathcal{K}(1^k), (m, \sigma) \leftarrow \mathcal{A}^{\mathcal{S}_{\mathsf{sk}}}(\mathsf{pk}) : \mathcal{V}(\mathsf{pk}, m, \sigma) = 1\ \right].$$

6.2.2 Signature et chiffrement

Nous cherchons à construire un padding unifié qui peut être utilisé en même temps et avec une même primitive pour le chiffrement et pour la signature. L'attaquant a pour objectif soit de construire une falsification existentielle (*i.e.* casser EUF) contre le schéma de signature, soit de casser la sécurité sémantique du schéma de chiffrement (*i.e.* casser IND). Pour y parvenir, il dispose de l'accès adaptatif aux oracles de signature et de déchiffrement. Il s'agit d'une combinaison des attaques contre le chiffrement et contre la signature, d'où la notation de l'attaque CMA + CCA.

6.2.3 Permutations sans griffe

Dans [71], Katz et Wang ont prouvé que, en utilisant des permutations à trappe induites par des permutations sans griffe (*claw-free permutations* en anglais), on peut obtenir une variante de FDH (en y ajoutant un bit supplémentaire) avec une réduction fine. Nous utilisons également cette technique pour notre construction. L'hypothèse de l'existence des permutations sans griffe semble être raisonnable. En effet, toute permutation à sens-unique aléatoirement auto-réductible peut être considérée comme une permutations sans griffe [42] et la quasi-totalité des exemples connus de permutations à sens-unique à trappe sont aléatoirement auto-réductibles.

Définition 46 (Permutations sans griffe) *Une famille de permutations sans griffe est un ensemble d'algorithmes* $\{\mathsf{Gen}; f_i; g_i | i \in I\}$*, tels que :*

- Gen *retourne un indice aléatoire* i *et une trappe* td.
- f_i, g_i *sont des permutations sur le même domaine* D_i.
- *il existe un algorithme d'échantillonnage efficace qui, pour chaque indice* i*, retourne une valeur aléatoire* $x \in D_i$.

– *étant donné la trappe* td, f_i^{-1} *(l'inverse de* f_i*) et* g_i^{-1} *(l'inverse de* g_i*) sont efficacement calculables.*

Une griffe est un couple (x_0, x_1) *tel que* $f(x_0) = g(x_1)$*. On dit que l'algorithme probabiliste* $\mathcal{A}$ (t, ϵ)*-casse une famille de permutations sans griffe si* $\mathcal{A}$*, exécuté en temps* t*, peut retourner une griffe avec une probabilité supérieure à* ϵ *:*

$$\Pr\left[\ (i, \mathsf{td}) \leftarrow \mathsf{Gen}(1^k), (x_0, x_1) \leftarrow \mathcal{A}(i)\ : f_i(x_0) = g_i(x_1)\ \right] \geq \epsilon.$$

Une famille de permutations sans griffe est dite (t, ϵ)*-sûre s'il n'existe pas d'algorithme qui puisse* (t, ϵ)*-casser cette famille.*

6.3 Padding optimal fondé sur des permutations aléatoires

Dans ce qui suit, nous proposons un padding universel dans le modèle de la permutation aléatoire, fondé sur notre construction présentée au chapitre précédent. Il est optimal pour signer et pour chiffrer, *i.e.* il utilise 82 bits de redondance pour signer et seulement 82 bits aléatoires pour chiffrer. Dans la section suivante, nous proposons une autre construction, fondée sur OAEP 3-tours [96], qui a été prouvée sûre dans le modèle de l'oracle aléatoire, et « presque » optimale (161 bits d'aléa au lieu de 82).

Le chiffrement et la signature utilisent une permutation $\mathcal{P}$, supposée ressembler à une permutation véritablement aléatoire. Notons k le paramètre de sécurité et $\varphi_{\mathsf{pk}} : \{0,1\}^n \rightarrow \{0,1\}^n$ une permutation à sens-unique à trappe (dont l'inverse est notée ψ_{sk}). Les messages à signer ou à chiffrer seront de taille $\ell = n-k-1$. Le symbole « $\|$ » dénote la concaténation de chaînes de bits. De plus, on identifie $\{0,1\}^k \times \{0,1\}^\ell \times \{0,1\}$ à $\{0,1\}^n$. Finalement, dans ce qui suit, $\mathsf{prf}()$ désigne une fonction pseudo-aléatoire.

6.3.1 Padding

Le padding est relativement simple, il prend comme entrées un bit γ, le message m et une donnée additionnelle r, et retourne $\mathsf{OPbP}(\gamma, m, r) = \mathcal{P}(\gamma \| m \| r) = t \| u$. L'inverse de cette opération est naturellement : $\mathsf{OPbP}^{-1}(t, u) = \mathcal{P}^{-1}(t \| u) = \gamma \| m \| r$.

Algorithme de chiffrement

L'espace des textes clairs est $\mathcal{M} = \{0,1\}^\ell$. Le chiffrement utilise une chaîne aléatoire $r \in \mathcal{R} = \{0,1\}^k$ et un bit aléatoire γ. Il retourne un texte chiffré c de $\{0,1\}^n$:

$$t \| u = \mathsf{OPbP}(\gamma, m, r) \qquad c = \varphi_{\mathsf{pk}}(t \| u).$$

Algorithme de déchiffrement

À partir du texte chiffré c, on calcule d'abord $t\|u = \psi_{\mathsf{sk}}(c)$, où $t \in \{0,1\}^k$ et $u \in \{0,1\}^{\ell+1}$, puis on retrouve le texte clair m contenu dans : $\gamma\|m\|r = \mathsf{OPbP}^{-1}(t,u)$.

Algorithme de signature

L'espace des textes clairs est $\mathcal{M} = \{0,1\}^\ell$, l'algorithme de signature retourne une signature σ de $\{0,1\}^n$: à partir du message $m \in \mathcal{M}$, on calcule $\gamma = \mathsf{prf}(m)$, puis $t\|u = \mathsf{OPbP}(\gamma, m, 0^k)$ et $\sigma = \psi_{\mathsf{sk}}(t\|u)$.

Algorithme de vérification

À partir de la signature σ, on calcule d'abord $t\|u = \varphi_{\mathsf{pk}}(\sigma)$, où $t \in \{0,1\}^k$ et $u \in \{0,1\}^{\ell+1}$, puis $\gamma\|m\|r = \mathsf{OPbP}^{-1}(t,u)$. Si $r = 0^k$, la vérification retourne « Correcte » et retrouve m, sinon elle retourne « Incorrecte ».

6.3.2 Analyse de sécurité

Dans le chapitre précédent, ce padding, sans le bit additionnel de Katz et de Wang, conduit à un schéma de chiffrement $\mathsf{IND/CCA}$ sûr dans le modèle de la permutation aléatoire. Cependant, ce bit additionnel ne rend que plus aléatoire la phase de chiffrement. Dans cette section, on étend ce résultat au scénario d'attaque $\mathsf{CMA} + \mathsf{CCA}$ ainsi qu'à la signature ($\mathsf{EUF/CMA} + \mathsf{CCA}$).

Théorème 47 *Considérons les attaquants $\mathcal{A}$ et $\mathcal{B}$ contre respectivement le chiffrement et la signature, selon une attaque à la fois à chiffrés choisis (avec accès à l'oracle de déchiffrement) et à messages choisis (avec accès à l'oracle de signature). Supposons qu'en temps τ et qu'après q_p, q_s, q_d requêtes respectivement aux oracles de permutation, de signature et de déchiffrement, $\mathcal{A}$ peut avoir un avantage ε_E pour casser la sécurité sémantique du schéma, ou $\mathcal{B}$ peut avoir un succès ε_S pour produire une falsification existentielle, alors la permutation φ_{pk} peut être inversée avec probabilité ε' en temps t' où :*

$$\varepsilon' \geq \varepsilon_E - \frac{(q_p + q_d + q_s + 1)^2}{2^{k+\ell+1}} - \frac{(q_d+1)^2}{2^\ell} - \frac{2q_p + q_d + q_s + 2}{2^k}$$

ou

$$\varepsilon' \geq \frac{1}{q_p + q_s + 1} \times \left(\varepsilon_S - \frac{(q_p + q_d + q_s + 1)^2}{2^{k+\ell+1}} - \frac{(q_d+1)^2}{2^\ell} - \frac{2q_p + q_d + q_s + 2}{2^k} \right).$$

En particulier, si la fonction φ_{pk} est induite par une famille de permutations sans griffe (t', ε')-sûre, la deuxième relation peut être ré-écrite par :

$$\varepsilon' \geq \frac{1}{2}\left(\varepsilon_S - \frac{(q_p + q_d + q_s + 1)^2}{2^{k+\ell+1}} - \frac{(q_d+1)^2}{2^\ell} - \frac{2q_p + q_d + q_s + 2}{2^k}\right).$$

où $t' \leq t + (q_p + q_d + q_s + 1)T_f$, et T_f est le temps pour une évaluation de φ_{pk}.

Preuve. On utilise la technique des jeux successifs pour simuler les oracles de permutation aléatoire, l'oracle de déchiffrement, le challengeur et l'oracle de signature.

JEU $\mathbf{J}_0$: Il s'agit du jeu réel d'attaque dans le modèle de la permutation aléatoire. L'attaquant a accès aux oracles de permutation aléatoire $\mathcal{P}$ et $\mathcal{P}^{-1}$, à l'oracle de déchiffrement $\mathcal{D}_{\mathsf{sk}}$ et à l'oracle de signature $\mathcal{S}_{\mathsf{sk}}$.

Concernant la sécurité du chiffrement, on considère un attaquant $A = (A_1, A_2)$ contre ce schéma selon une attaque IND-CMA + CCA. Après avoir pris connaissance de la clef publique (la description de la fonction φ_{pk}), A_1 retourne deux textes clairs (m_0, m_1). Puis, le challengeur fabrique un challenge : un bit aléatoire b est choisi et le challenge $c^\star$ de $m^\star = m_b$ est donné par : $c^\star = \mathcal{E}(\gamma^\star, m_b, r^\star) = \varphi_{\mathsf{pk}}(\mathcal{P}(\gamma^\star, m_b, r^\star))$, où $r^\star \xleftarrow{R} \{0,1\}^k$, et $\gamma^\star$ est un bit aléatoire. Après avoir reçu le challenge $c^\star$, A_2 retourne un bit b'. On note Dist_0 l´événement $b' = b$ et Dist_n dans les jeux $\mathbf{J}_n$ ci-dessous.

En ce qui concerne la sécurité de la signature, on considère l'attaquant $\mathcal{B}$ contre ce schéma selon une attaque EUF-CMA + CCA. $\mathcal{B}$ retourne une falsification et on vérifie si elle est valide ou pas. On note l'événement où cette falsification est valide Forge_0 et Forge_n dans les jeux $\mathbf{J}_n$ ci-dessous.

Remarquons que l'attaquant a accès aux oracles de déchiffrement $\mathcal{D}_{\mathsf{sk}}$ et de signature $\mathcal{S}_{\mathsf{sk}}$ à toutes les étapes de l'attaque. Remarquons aussi que si l'attaquant soumet q_d, q_s et q_p requêtes respectivement aux oracles de déchiffrement, de signature et de permutation aléatoire, le nombre maximal de requêtes soumises à l'oracle de permutation aléatoire est $q_d + q_s + q_p + 2$. En effet, chaque requête de déchiffrement ou de signature peut faire une requête de permutation aléatoire, de même que l'étape de vérification ou l'étape de génération du challenge.

Par définition :

$$\begin{aligned} \varepsilon_E &= \mathsf{Adv}^{\mathsf{ind/cma+cca}}_{\mathsf{OPbP}}(\mathcal{A}) = 2\Pr[\mathsf{Dist}_0] - 1 \\ \varepsilon_S &= \mathsf{Succ}^{\mathsf{euf/cma+cca}}_{\mathsf{OPbP}}(\mathcal{B}) = \Pr[\mathsf{Forge}_0]. \end{aligned}$$

JEU $\mathbf{J}_1$: Une simulation parfaite du jeu réel est décrite dans la Figure 6.1. En effet, la règle $\mathsf{Chal}^{(1)}$ simule parfaitement la génération du challenge $c^\star$. Dans ce qui suit, on simule les permutations aléatoires $\mathcal{P}$ et $\mathcal{P}^{-1}$ grâce à la liste $\mathcal{P}$-List, à une permutation aléatoire P et à son inverse P^{-1}.

Oracle $\mathcal{P}$	Requête $\mathcal{P}(\gamma, m, r)$: la réponse est p, où ►**Règle** $\mathsf{EvalP}^{(0)}$ $\quad p = P(\gamma, m, r)$. En outre, si (γ, m, r) est une requête directe de l'attaquant à $\mathcal{P}$, ajouter $(\gamma, m, r, p, \perp, \varphi_{\mathsf{pk}}(p))$ à $\mathcal{P}$-List.
Oracle $\mathcal{P}^{-1}$	Requête $\mathcal{P}^{-1}(p)$: la réponse est (γ, m, r), où ►**Règle** $\mathsf{InvP}^{(0)}$ $\quad (\gamma, m, r) = P^{-1}(p)$. En outre, si p est une requête directe de l'attaquant à $\mathcal{P}^{-1}$, ajouter $(\gamma, m, r, p, \perp, \varphi_{\mathsf{pk}}(p))$ à $\mathcal{P}$-List.
Oracle $\mathcal{D}$	Requête $\mathcal{D}_{\mathsf{sk}}(c)$: la réponse est m : ►**Règle** $\mathsf{Decrypt}^{(0)}$ $\quad p = \psi_{\mathsf{sk}}(c)$, et $(\gamma, m, r) = \mathcal{P}^{-1}(p)$. Ajouter $(\gamma, m, r, \perp, \perp, c)$ à $\mathcal{P}$-List.
Oracle $\mathcal{S}$	Requête $\mathcal{S}_{\mathsf{sk}}(m)$: la réponse est σ : calculer $\gamma = \mathsf{prf}(m)$, puis soumettre $(\gamma, m, 0^k)$ à l'oracle de permutation EvalP : $p = \mathcal{P}(\gamma, m, 0^k)$. ►**Règle** $\mathcal{S}^{(0)}$ $\quad$ Calculer $\sigma = \psi_{\mathsf{sk}}(p)$. Ajouter $(\gamma, m, 0^k, p, \sigma, \varphi_{\mathsf{pk}}(p))$ à $\mathcal{P}$-List.
Challengeur	Pour deux messages (m_0, m_1) : choisir aléatoirement un bit b et définir $m^\star = m_b$; choisir aléatoirement $r^\star$ et répondre $c^\star$ où ►**Règle** $\mathsf{Chal}^{(0)}$ $\quad p^\star = \mathcal{P}(\gamma^\star, m^\star, r^\star)$; $c^\star = \varphi_{\mathsf{pk}}(p^\star)$. ►**Règle** $\mathsf{ChalAdd}^{(0)}$ $\quad$ Ajouter $(\gamma^\star, m^\star, r^\star, \perp, \perp, c^\star)$ à $\mathcal{P}$-List.
Oracle $\mathcal{V}$	Vérification de la falsification (σ) de l'attaquant. D'abord, calculer $t\|u = \varphi_{\mathsf{pk}}(\sigma)$. Ensuite, faire une requête à l'oracle de permutation $(\gamma, m, r) = \mathcal{P}^{-1}(t\|u)$. Finalement, vérifier si $r = 0^k$, auquel cas, la falsification est une signature valide de m.

FIG. 6.1 – Simulation dans le jeu $\mathbf{J}_1$

Il convient de noter que $\mathcal{P}$-List contient des éléments de la forme (γ, m, r, p, s, c) : p représente la valeur de $\mathcal{P}(\gamma, m, r)$; c , la valeur de $\varphi_{\mathsf{pk}}(p)$; et s , la valeur de $\psi_{\mathsf{sk}}(p)$, *i.e.* $p = \varphi_{\mathsf{pk}}(s)$. Dans l'organisation de $\mathcal{P}$-List, lors que l'on ajoute (γ, m, r, p, s, c) à $\mathcal{P}$-List, s'il y a déjà un élément de la forme $(\gamma, m, r, \alpha, \beta, y)$ dans $\mathcal{P}$-List, avec $\alpha = \perp$ ou $\beta = \perp$, on remplace cet élément par (m, r, p, s, c).

Maintenant, on note Δ_n la distance statistique entre les distributions de la vue de l'attaquant dans les jeux $\mathbf{J}_n$ et $\mathbf{J}_{n-1}$. On voit facilement que : $\Delta_1 = 0$.

JEU $\mathbf{J}_2$: Dans ce jeu, on modifie la simulation des oracles $\mathcal{P}$ et $\mathcal{P}^{-1}$ de telle sorte que chaque requête ne soit soumise au maximum qu'une fois à la permutation aléatoire P ou à son inverse P^{-1} :

►**Règle** $\mathsf{EvalP}^{(2)}$

- Si $(\gamma, m, r, \alpha, \beta, c)$ se trouve dans $\mathcal{P}$-List :
  - si $\alpha \neq \perp$, $p = \alpha$;
  - sinon, Terminer.
- Sinon, $p = P(\gamma, m, r)$.

►**Règle** $\mathsf{InvP}^{(2)}$

Calculer $c = \varphi_{\mathsf{pk}}(p)$ et vérifier si $(\gamma, m, r, \alpha, \beta, c)$ est dans $\mathcal{P}$-List :
- si oui, (γ, m, r) est déjà défini ;
- sinon $(\gamma, m, r) = P^{-1}(p)$.

On est amené à donner une réponse incorrecte lors de la simulation de l'oracle $\mathcal{P}$ quand il existe un élément $(\gamma, m, r, \alpha, \beta, c)$ dans $\mathcal{P}$-List et $\alpha = \perp$. Dans ce cas, on met fin au jeu mais la réponse correcte devrait être $\psi_{\mathsf{sk}}(c)$. On note cet événement BadP_2 et BadP_n dans les jeux $\mathbf{J}_n$ ci-dessous. À l'exception de cet événement, les jeux $\mathbf{J}_2$ et $\mathbf{J}_1$ sont parfaitement indistinguables :

$$\Delta_2 \leq \Pr[\mathsf{BadP}_2].$$

JEU $\mathbf{J}_3$: On modifie la simulation des oracles $\mathcal{P}$ et $\mathcal{P}^{-1}$. On n'utilise plus la permutation aléatoire P et son inverse P^{-1} mais retourne simplement une valeur aléatoire pour chaque nouvelle requête : à une nouvelle requête (γ, m, r) à l'oracle $\mathcal{P}$, on répond un aléa p, et à une nouvelle requête p à l'oracle $\mathcal{P}^{-1}(p)$, on répond aléatoirement (γ, m, r).

►**Règle** $\mathsf{EvalP}^{(3)}$

- Si $(\gamma, m, r, \alpha, \beta, c)$ se trouve dans $\mathcal{P}$-List :
  - si $\alpha \neq \perp$, $p = \alpha$;
  - sinon, Terminer.
- Sinon, $p \xleftarrow{R} \{0,1\}^n$.

►**Règle** $\mathsf{InvP}^{(3)}$

Calculer $c = \varphi_{\mathsf{pk}}(p)$ et vérifier si $(\gamma, m, r, \alpha, \beta, c)$ est dans $\mathcal{P}$-List :
- si oui, (γ, m, r) est défini ;
- sinon on choisit $(\gamma, m, r) \xleftarrow{R} \{0,1\}^n$.

Les jeux $\mathbf{J}_3$ et $\mathbf{J}_2$ sont parfaitement indistinguables sauf s'il y a une collision sur (γ, m, r) ou p dans $\mathcal{P}$-List. On note CollP_3 cet événement. On a :

$$\Delta_3 \leq \Pr[\mathsf{CollP}_3].$$

Comme il y a au plus $q_p + q_d + q_s + 1$ éléments dans $\mathcal{P}$-List, la probabilité d'occurrence d'une collision est $(q_p + q_d + q_s + 1)^2/2^n$:

$$\Pr[\mathsf{CollP}_3] \leq \frac{(q_p + q_d + +q_s + 1)^2}{2^{k+\ell+1}}.$$

Jeu $\mathbf{J}_4$: On peut maintenant simuler l'oracle de déchiffrement $\mathcal{D}_{\mathsf{sk}}$ sans utiliser la fonction ψ_{sk} :

►**Règle** $\mathsf{Decrypt}^{(4)}$

Vérifier s'il y a $(\gamma, m, r, \alpha, \beta, c)$ dans $\mathcal{P}$-List :

1. si oui, (γ, m, r) est défini,
2. sinon, on choisit $(\gamma, m, r) \stackrel{R}{\leftarrow} \{0,1\}^n$.

Les jeux $\mathbf{J}_4$ et $\mathbf{J}_3$ sont parfaitement indistinguables car dans le deuxième cas, on fait exactement comme si $\mathcal{P}^{-1}$ était simulé par la règle $\mathsf{InvP}^{(3)}$ dans le jeu précédent, y compris l'ajout à la $\mathcal{P}$-List :

$$\Delta_4 = 0.$$

Jeu $\mathbf{J}_5$: On considère les éléments de la forme $(\gamma, m, r, \perp, \beta, c)$ dans $\mathcal{P}$-List (il y en a au plus q_d+1). S'il y a une collision sur m, on met fin au jeu. On note CollM_5 cet événement :

$$\Delta_5 \leq \Pr[\mathsf{CollM}_5].$$

On remarque que m est aléatoirement choisi pour tout $(\gamma, m, r, \perp, \beta, c)$ sauf si $m = m^\star$. Par conséquent, la probabilité d'occurrence d'une collision sur m est majorée par $(q_d+1)^2/2^\ell$:

$$\Pr[\mathsf{CollM}_5] \leq \frac{(q_d + 1)^2}{2^\ell}.$$

Dans le cas où il n'y a pas de collision, pour chaque m, il existe au plus une valeur de r (et une valeur de c) telle(s) que $(\gamma, m, r, \perp, \beta, c) \in \mathcal{P}$-List. Donc, on peut facilement voir que :

$$\Pr[\mathsf{BadP}_5] \leq \frac{q_p + q_s + 1}{2^k}.$$

Jeu $\mathbf{J}_6$: On simule l'oracle de permutation pour ne plus avoir à utiliser la fonction ψ_{sk} dans la simulation de l'oracle de signature. Pour ce faire, on fait appel à la fonction $\varphi_{\mathsf{pk}}(s)$ lors de la simulation de l'oracle de permutation :

►**Règle** $\mathsf{EvalP}^{(6)}$

- Si $(\gamma, m, r, \alpha, \beta, c)$ se trouve dans $\mathcal{P}$-List :
  - si $\alpha \neq \perp$, $p = \alpha$;
  - sinon, Terminer.
- Sinon, choisir $s \xleftarrow{R} \{0,1\}^n$ et calculer $p = \varphi_{\mathsf{pk}}(s)$. Ajouter $(\gamma, m, r, p, s, \varphi_{\mathsf{pk}}(p))$ à $\mathcal{P}$-List.

Comme φ_{pk} est une permutation, p est un aléa et par conséquent, les jeux $\mathbf{J}_6$ et $\mathbf{J}_5$ sont parfaitement indistinguables :

$$\Delta_6 = 0.$$

JEU $\mathbf{J}_7$: Dans ce jeu, on continue à simuler l'oracle $\mathcal{P}^{-1}$ afin qu'il ne retourne pas d'élément de la forme $(\gamma, m, 0^k)$. En effet, si l'attaquant soumet $a = \mathcal{P}^{-1}(p)$, il sait que $\mathcal{P}(a) = p$. Par conséquent, si a est de la forme $\gamma \| m_0 \| 0^k$, on ne peut pas simuler la signature de m_0 sans utiliser ψ_{sk} car on ne connaît pas la valeur de s telle que $\mathcal{P}(a) = \varphi_{\mathsf{pk}}(s)$.

►**Règle** $\mathsf{InvP}^{(7)}$

Calculer $c = \varphi_{\mathsf{pk}}(p)$ et vérifier si $(\gamma, m, r, \alpha, \beta, c)$ est dans $\mathcal{P}$-List :
- si oui, (γ, m, r) est défini ;
- sinon on choisit $(\gamma, m, r) \xleftarrow{R} \{0,1\}^n$. Si $r = 0^k$, Terminer.

Pour chaque requête à $\mathcal{P}^{-1}$, la probabilité que l'on termine le jeu est égale à $1/2^k$:

$$\Delta_7 \leq (q_p + q_d + 1)/2^k.$$

JEU $\mathbf{J}_8$: Toutes les valeurs $\mathcal{P}(\gamma, m, 0^k)$ dont l'attaquant dispose sont des sorties de l'oracle $\mathcal{P}$. Par conséquent, $\mathcal{P}(\gamma, m, 0^k) = \varphi_{\mathsf{pk}}(s)$ et on peut donc simuler l'oracle de signature sans utiliser ψ_{sk} :

►**Règle** $\mathcal{S}_{\mathsf{sk}}{}^{(8)}$

Consulter $(\gamma, m, 0^k, p, s, c)$ dans $\mathcal{P}$-List, et définir $\sigma = s$.

Les jeux $\mathbf{J}_8$ et $\mathbf{J}_7$ sont parfaitement indistinguables :

$$\Delta_8 = 0.$$

On a simulé toutes les requêtes que l'attaquant soumet aux oracles de permutation, de déchiffrement et de signature. Ces simulations, décrites dans les figures 6.2 et 6.3, sont indépendantes de l'objectif de l'attaquant. Dans la suite, en fonction de son objectif (contre le chiffrement ou contre la signature), on complétera la réduction au problème d'inverser la fonction φ_{sk} pour une instance y donnée.

Attaque contre le chiffrement

JEU $\mathbf{J}_{8.1}$: On enlève l'élément $(\gamma^\star, m^\star, r^\star, \perp, \perp, c^\star)$ de $\mathcal{P}$-List lors de la génération du challenge.

Challengeur	Pour deux messages (m_0, m_1) : choisir aléatoirement un bit b et définir $m^\star = m_b$; choisir aléatoirement $r^\star$ et répondre $c^\star$ où ►**Règle** $\mathsf{Chal}^{(8)}$ $\quad p^\star = \mathcal{P}(\gamma^\star, m^\star, r^\star)$; $c^\star = \varphi_{\mathsf{pk}}(p^\star)$. ►**Règle** $\mathsf{ChalAdd}^{(8)}$ $\quad$ ajouter $(\gamma^\star, m^\star, r^\star, \perp, \perp, c^\star)$ à $\mathcal{P}$-List.
Oracle $\mathcal{V}$	Vérification de la falsification (σ) de l'attaquant. D'abord, calculer $t\|\|u = \varphi_{\mathsf{pk}}(\sigma)$. Ensuite, faire une requête à l'oracle de permutation $(\gamma, m, r) = \mathcal{P}^{-1}(t\|\|u)$. Finalement, vérifier si $r = 0^k$, dans ce cas, la falsification est une signature valide de m.

FIG. 6.2 – Simulation dans le Jeu $\mathbf{J}_8$

►**Règle** $\mathsf{ChalAdd}^{(8.1)}$

| Ne rien faire.

Les jeux $\mathbf{J}_{8.1}$ et $\mathbf{J}_8$ sont parfaitement indistinguables sauf si $(\gamma^\star, m^\star, r^\star)$ a été soumise à $\mathcal{P}$ ou $p^\star = \psi_{\mathsf{sk}}(c^\star)$ a été soumise à $\mathcal{P}^{-1}$. Le premier événement est compris dans l'événement $\mathsf{BadP}_{8.1}$, qui est déjà exclu. On note $\mathsf{AskInvP}_{8.1}$ le deuxième événement :

$$\Delta_{8.1} \leq \Pr[\mathsf{AskInvP}_{8.1}].$$

Dans ce jeu, $(\gamma^\star, m^\star, r^\star, \perp, \perp, c^\star)$ n'est plus dans $\mathcal{P}$-List, l'attaquant n'a aucun avantage pour deviner b : $\Pr[\mathsf{Dist}_{8.1}] = \frac{1}{2}$.

JEU $\mathbf{J}_{8.2}$: Au lieu de choisir $c^\star = \varphi_{\mathsf{pk}}(p^\star)$, on choisit $c^\star = y$, uniformément aléatoire.

►**Règle** $\mathsf{Chal}^{(8.2)}$

| $c^\star = y$.

Alors, on définit implicitement $p^\star = \psi_{\mathsf{sk}}(y)$. Comme $(\gamma^\star, m^\star, r^\star, \perp, \perp, c^\star)$ n'est plus utilisé dans la simulation, les jeux $\mathbf{J}_{8.2}$ et $\mathbf{J}_{8.1}$ sont parfaitement indistinguables : $\Delta_{8.2} = 0$.

Finalement, on peut facilement calculer $\psi_{\mathsf{sk}}(y)$ lorsque l'événement $\mathsf{AskInvP}_{8.2}$ se produit : en consultant dans $\mathcal{P}$-List (qui contient au plus $q_p + q_d + q_s + 1$ éléments), on peut extraire p tel que $y = \varphi_{\mathsf{pk}}(p)$. Alors :

$$\Pr[\mathsf{AskInvP}_{8.2}] \leq \mathsf{Succ}^{\mathsf{ow}}_{\varphi}(t'),$$

où T_φ est le temps nécessaire pour une évaluation d'une fonction φ_{pk}, et $t' \leq t + (q_p + q_d + q_s + 1) \times T_\varphi$, le temps de la simulation de ce jeu.

Attaque contre la signature (cas général)

JEU $\mathbf{J}_{8.1}$: Dans ce jeu, on énumère les requêtes de la forme $(\gamma, \star, 0^k)$ à l'oracle de permutation. Il s'agit des requêtes qui seront utilisées pour la signature. Pour ce faire, on

Oracle $\mathcal{P}$

Requête $\mathcal{P}(\gamma, m, r)$: la réponse est p, où :

►**Règle** $\mathsf{EvalP}^{(8)}$

- Si $(\gamma, m, r, \alpha, \beta, c)$ se trouve dans $\mathcal{P}$-List :
  - si $\alpha \neq \perp$, $p = \alpha$;
  - sinon, Terminer.
- Sinon, choisir $s \xleftarrow{R} \{0,1\}^n$, calculer $p = \varphi_{\mathsf{pk}}(s)$ et ajouter $(\gamma, m, r, p, s, \varphi_{\mathsf{pk}}(p))$ à $\mathcal{P}$-List.

En outre, si (γ, m, r) est une requête directe de l'attaquant à $\mathcal{P}$, ajouter $(\gamma, m, r, p, \perp, \varphi_{\mathsf{pk}}(p))$ à $\mathcal{P}$-List.

Oracle $\mathcal{P}^{-1}$

Requête $\mathcal{P}^{-1}(p)$: la réponse est (γ, m, r) :

►**Règle** $\mathsf{InvP}^{(8)}$

Calculer $c = \varphi_{\mathsf{pk}}(p)$ et vérifier si $(\gamma, m, r, \alpha, \beta, c)$ se trouve dans $\mathcal{P}$-List :
- si oui, (γ, m, r) est défini ;
- sinon, choisir $(\gamma, m, r) \xleftarrow{R} \{0,1\}^n$. Si $r = 0^k$, Terminer.

En outre, si p est une requête directe de l'attaquant à $\mathcal{P}^{-1}$, ajouter $(\gamma, m, r, p, \perp, \varphi_{\mathsf{pk}}(p))$ à $\mathcal{P}$-List.

Oracle $\mathcal{D}$

Requête $\mathcal{D}_{\mathsf{sk}}(c)$: la réponse est m :

►**Règle** $\mathsf{Decrypt}^{(8)}$

Vérifier si $(\gamma, m, r, \alpha, \beta, c)$ se trouve dans $\mathcal{P}$-List :
- si oui, (γ, m, r) est défini,
- sinon, choisir $(\gamma, m, r) \xleftarrow{R} \{0,1\}^n$.

Ajouter $(\gamma, m, r, \perp, \perp, c)$ à $\mathcal{P}$-List.

Oracle $\mathcal{S}$

Requête $\mathcal{S}_{\mathsf{sk}}(m)$: la réponse est σ : calculer $\gamma = \mathsf{prf}(m)$, puis soumettre $(\gamma, m, 0^k)$ à l'oracle de permutation EvalP : $p = \mathcal{P}(\gamma, m, 0^k)$.

►**Règle** $\mathcal{S}^{(8)}$

Consulter l'élément $(\gamma, m, 0^k, p, s, c)$ dans $\mathcal{P}$-List, et définir $\sigma = s$.

FIG. 6.3 – Simulation dans le Jeu $\mathbf{J}_8$

définit un compteur ν, initialisé à 0.

►**Règle** $\mathsf{EvalP}^{(8.1)}$

- Si $(\gamma, m, r, \alpha, \beta, c)$ se trouve dans $\mathcal{P}$-List :
  - si $\alpha \neq \perp$, $p = \alpha$;
  - sinon, Terminer.
- Sinon,
  - si $r = 0^k$, incrémenter ν ;
  - choisir $s \xleftarrow{R} \{0,1\}^n$ et calculer $p = \varphi_{\mathsf{pk}}(s)$. Ajouter $(\gamma, m, r, p, s, \varphi_{\mathsf{pk}}(p))$ à $\mathcal{P}$-List.

Évidemment, ce jeu est indistinguable du jeu $\mathbf{J}_8$: $\Delta_{8.1} = 0$.

Jeu $\mathbf{J}_{8.2}$: Comme le processus de vérification est compris dans le jeu d'attaque, la falsification est nécessairement soumise à l'oracle de permutation EvalP. On devine l'indice ν_0 de la première occurrence de cette requête. On considère seulement le cas où on le devine correctement :

$$\Pr[\mathsf{Forge}_{8.2}] \;\geq\; \Pr[\mathsf{Forge}_{8.1}]/(q_p + q_s + 1).$$

Jeu $\mathbf{J}_{8.3}$: On peut maintenant intégrer le challenge y dans la simulation de l'oracle de permutation. Grâce à cette méthode, on va extraire la pré-image x. L'idée est de répondre y à la ν_0-ième requête de l'oracle de permutation :

►**Règle** $\mathsf{EvalP}^{(8.3)}$

- Si $(\gamma, m, r, \alpha, \beta, c)$ se trouve dans $\mathcal{P}$-List :
  - si $\alpha \neq \perp$, $p = \alpha$;
  - sinon, Terminer.
- Sinon,
  - si $r = 0^k$, incrémenter ν ;
  - si $\nu \neq \nu_0$ ou si $r \neq 0^k$, choisir $s \xleftarrow{R} \{0,1\}^n$ et calculer $p = \varphi_{\mathsf{pk}}(s)$;
  - si $\nu = \nu_0$ et $r = 0^k$, définir $p = y$ et $s \xleftarrow{R} \{0,1\}^n$.
  - Ajouter $(\gamma, m, r, p, s, \varphi_{\mathsf{pk}}(p))$ à $\mathcal{P}$-List.

Comme l'instance y est aléatoirement choisie, les jeux $\mathbf{J}_{8.3}$ et $\mathbf{J}_{8.2}$ sont parfaitement indistinguables :

$$\Delta_{8.3} = 0.$$

Dans ce jeu, une falsification valide de signature conduit à calculer la pré-image de y :

$$\Pr[\mathsf{Forge}_{8.3}] = \mathsf{Succ}^{\mathsf{ow}}_{\varphi}(t + (q_p + q_d + q_s + 1)T_\varphi).$$

Cette égalité conclut la deuxième partie de la preuve.

Attaque contre la signature (avec une permutation sans griffe (t, ε')-sûre)

Dans le cas général, le bit γ peut en effet être supprimé car il est inutile. Dans le cas de permutation sans griffe $(\varphi_{\mathsf{pk}}, \lambda_{\mathsf{pk}})$ (t, ε')-sûres, l'idée est d'utiliser φ_{pk} à la sortie d'OPbP

pour une valeur du bit γ, et d'utiliser λ_{pk} pour l'autre valeur de γ. Comme la valeur de γ est imprévisible pour l'attaquant, la falsification conduit, avec une probabilité de $\frac{1}{2}$, à une griffe.

JEU $\mathbf{J}_{8.1}$: On exploite maintenant le bit γ pour la simulation de l'oracle de permutation, de la même manière que celle proposée par Katz et Wang [71].

▶**Règle** $\mathsf{EvalP}^{(8.1)}$
- Si $(\gamma, m, r, \alpha, \beta, c)$ se trouve dans $\mathcal{P}$-List :
  - si $\alpha \neq \perp$, $p = \alpha$;
  - sinon, Terminer.
- Sinon,
  - si $r \neq 0^k$ ou $\gamma = \mathsf{prf}(m)$, choisir $s \xleftarrow{R} \{0,1\}^n$ et calculer $p = \varphi_{\mathsf{pk}}(s)$.
  - si $r = 0^k$ ou $\gamma \neq \mathsf{prf}(m)$, choisir $s \xleftarrow{R} \{0,1\}^n$et calculer $p = \lambda_{\mathsf{pk}}(s)$.
  - Ajouter $(\gamma, m, r, p, s, \varphi_{\mathsf{pk}}(p))$ à $\mathcal{P}$-List.

Comme s est aléatoirement choisie, $\lambda_{\mathsf{pk}}(s)$ l'est aussi. Par conséquent, les jeux $\mathbf{J}_{8.1}$ et $\mathbf{J}_8$ sont parfaitement indistinguables :

$$\Delta_{8.1} = 0.$$

En utilisant les mêmes arguments que ceux présentés dans [71], on peut montrer qu'une falsification correcte de signature conduit à une griffe avec probabilité $\frac{1}{2}$. En effet, supposons que l'attaquant peut forger une signature $(\tilde{m}, \tilde{\sigma})$, où $(\tilde{m}, 0^k)$ a été soumise à l'oracle de permutation $\mathcal{P}$ lors d'une requête de permutation ou de l'étape de vérification. Le bit $b_{\tilde{m}} = \mathsf{prf}(\tilde{m})$ est un bit aléatoire et imprévisible pour l'attaquant. Alors, avec probabilité $\frac{1}{2}$, il existe un élément $(\tilde{m}, \tilde{r}, \tilde{p} = \lambda_{\mathsf{pk}}(\tilde{s}), \tilde{s}, \varphi_{\mathsf{pk}}(\tilde{p}))$ dans $\mathcal{P}$-List. Dans ce cas, on peut produire une griffe $\varphi_{\mathsf{pk}}(\tilde{\sigma}) = \lambda_{\mathsf{pk}}(\tilde{s})$. □

6.3.3 Proposition de taille pour les paramètres

Un schéma atteint le niveau de sécurité 2^κ si le rapport entre le temps t de fonctionnement de l'attaquant et sa probabilité de succès ε est au moins 2^κ. Il s'agit d'une approximation du temp pour obtenir un succès. Ainsi, on voudrait avoir $\varepsilon/t \leq 2^{-\kappa}$, avec la borne usuelle de sécurité $\kappa = 80$.

Tout d'abord, on cherche à simplifier le résultat ci-dessus. En effet, la taille ℓ du message est normalement significativement plus large que la taille k de l'aléa/la redondance : la quantité $Q/2^\ell$, voire $Q^2/2^\ell$, peut être ignorée en comparaison de $Q/2^k$ (où Q est le nombre total de requêtes, et donc Q est borné par 2^{80}). Par conséquent, le coût de la réduction dans le théorème 47 se simplifie :

$$\begin{aligned} & \frac{\varepsilon_E}{t} \leq \frac{\varepsilon'}{t} + \frac{2}{2^k} \\ \text{et} \quad & \frac{\varepsilon_S}{t} \leq \frac{Q\varepsilon'}{t} + \frac{2}{2^k} \text{ dans le cas général} \\ & \leq \frac{2\varepsilon'}{t} + \frac{2}{2^k} \text{ si la fonction } \varphi_{\mathsf{pk}} \text{ est induite par une permutation sans griffe.} \end{aligned}$$

Dans le cas particulier où la fonction φ_{pk} est induite par une permutation sans griffe (cas le plus courant où on utilise le RSA), il faut que la taille du message (*i.e.* module RSA) soit suffisamment grande pour que ε'/t soit inférieur à 2^{-82}. Grâce à l'estimation de Lenstra-Verheul [77], on peut utiliser un module RSA de 1024 bits. Avec seulement 82 bits de redondance, nous obtenons le même niveau de sécurité que RSA-PSS [10], et une meilleure bande passante. Concernant la sécurité du chiffrement, le résultat du chapitre précédent reste valable : 82 bits d'aléa sont convenables pour la sécurité sémantique, même selon des attaques $\mathsf{CMA} + \mathsf{CCA}$.

Dans le cas général, il nous faut un paramètre de sécurité et un message de taille relativement grande pour que ε'/t soit inférieur à 2^{-161}. Alors, si le niveau de sécurité de la fonction φ est 2^{161}, on peut choisir l'« overhead » $k = 82$ qui jouera le rôle de la redondance pour la signature et celui d'aléa pour le chiffrement.

6.4 OAEP 3-tours pour le chiffrement et la signature

6.4.1 Description

Afin de construire un schéma dans le modèle de l'oracle aléatoire [9], nous considérons la construction d'OAEP 3-tours présentée dans le chapitre précédent.

Le chiffrement et la signature utilisent trois fonctions de hachage : $\mathcal{F}$, $\mathcal{G}$ et $\mathcal{H}$, supposées ressembler à des oracles aléatoires dans les analyses de sécurité. Les paramètres de sécurité satisfont $n = k + \ell + 1$:

$$\mathcal{F} : \{0,1\}^k \to \{0,1\}^{\ell+1} \qquad \mathcal{G} : \{0,1\}^{\ell+1} \to \{0,1\}^k \qquad \mathcal{H} : \{0,1\}^k \to \{0,1\}^{\ell+1}.$$

Le chiffrement et la signature utilisent aussi une famille de permutations à sens-unique à trappe (φ_{pk}), sur l'espace $\{0,1\}^n$, dont les inverses sont les ψ_{sk}, où sk est la clef secrète (*i.e.* la trappe) associée à la clef publique pk. De plus, on identifie $\{0,1\} \times \{0,1\}^k \times \{0,1\}^\ell$ à $\{0,1\}^n$. Finalement, dans ce qui suit, $\mathsf{prf}()$ désigne une fonction pseudo-aléatoire.

Padding $\mathsf{OAEP3r}$ et unpadding $\mathsf{OAEP3r}^{-1}$

– $\mathsf{OAEP3r}(\gamma, m, r)$:

$$s = (\gamma \| m) \oplus \mathcal{F}(r) \qquad t = r \oplus \mathcal{G}(s) \qquad u = s \oplus \mathcal{H}(t)$$

$$\mathsf{OAEP3r}(\gamma, m, r) = t\|u.$$

– $\mathsf{OAEP3r}^{-1}(t, u)$:

$$s = u \oplus \mathcal{H}(t) \qquad r = t \oplus \mathcal{G}(s) \qquad \gamma\|m = s \oplus \mathcal{F}(r)$$

$$\mathsf{OAEP3r}^{-1}(t, u) = \gamma\|m\|r.$$

Algorithme de chiffrement

L'espace des textes clairs est $\mathcal{M} = \{0,1\}^\ell$. Le chiffrement utilise une chaîne aléatoire $r \in \mathcal{R} = \{0,1\}^k$ et un bit aléatoire γ. À partir du texte clair $m \in \mathcal{M}$, il retourne un texte chiffré c de $\{0,1\}^n$:

$$t\|u = \mathsf{OAEP3r}(\gamma, m, r) \qquad c = \varphi_{\mathsf{pk}}(t\|u).$$

Algorithme de déchiffrement

À partir du texte chiffré c, on calcule d'abord $t\|u = \psi_{\mathsf{sk}}(c)$, où $t \in \{0,1\}^k$ et $u \in \{0,1\}^{\ell+1}$, puis on retourne le texte clair m en calculant : $\gamma\|m\|r = \mathsf{OAEP3r}^{-1}(t, u)$.

Algorithme de signature

L'espace des textes clairs est $\mathcal{M} = \{0,1\}^\ell$, l'algorithme de signature retourne une signature σ de $\{0,1\}^n$: à partir du message $m \in \mathcal{M}$, on calcule $\gamma = \mathsf{prf}(m)$, puis $t\|u = \mathsf{OAEP3r}(\gamma, m, 0^k)$ et $\sigma = \psi_{\mathsf{sk}}(t\|u)$

Algorithme de vérification

À partir de la signature σ, on calcule d'abord $t\|u = \varphi_{\mathsf{pk}}(\sigma)$, où $t \in \{0,1\}^k$ et $u \in \{0,1\}^{\ell+1}$, puis $\gamma\|m\|r = \mathsf{OAEP3r}^{-1}(t, u)$. Si $r = 0^k$, la vérification retourne « Correcte » et retrouve m, sinon elle retourne « Incorrecte ».

6.4.2 Résultat de sécurité

Théorème 48 *Considérons les attaquants $\mathcal{A}$ et $\mathcal{B}$ contre respectivement le chiffrement et la signature selon une attaque à la fois à chiffrés choisis et à messages choisis. Supposons qu'en temps τ et qu'après q_f, q_g, q_h, q_s, q_d requêtes respectivement aux oracles $\mathcal{F}$, $\mathcal{G}$, $\mathcal{H}$, de signature et de déchiffrement, $\mathcal{A}$ peut avoir un avantage ε_E pour casser la sécurité sémantique du schéma, ou $\mathcal{B}$ peut avoir un succès ε_S pour produire une falsification existentielle, alors la permutation φ_{pk} peut être inversée avec probabilité ε' en temps τ' où :*

$$\begin{aligned}\varepsilon' &\geq \varepsilon_E - \left(q_d^2 \times \left(\frac{1}{2^{\ell+1}} + \frac{6}{2^k}\right) + \frac{4q_dq_g + q_g}{2^{\ell+1}} + \frac{5q_dq_f + q_gq_h + q_f + q_d}{2^k}\right)\\ \textit{ou}\\ \varepsilon' &\geq \frac{1}{q_g + q_s + 1} \times \\ &\quad \left(\varepsilon_S - \left(q_d^2 \times \left(\frac{1}{2^{\ell+1}} + \frac{6}{2^k}\right) + \frac{4q_dq_g + q_g}{2^{\ell+1}} + \frac{5q_dq_f + q_gq_h + q_f + q_d}{2^k}\right)\right).\end{aligned}$$

En particulier, si la fonction φ_{pk} est induite par une famille de permutations sans griffe et (τ, ε')-sûre, la deuxième relation peut être ré-écrite par :

$$\varepsilon' \geq \frac{1}{2} \times \left(\varepsilon_S - \left(q_d^2 \times \left(\frac{1}{2^{\ell+1}} + \frac{6}{2^k}\right) + \frac{4q_dq_g}{2^{\ell+1}} + \frac{5q_dq_f}{2^k} + \frac{q_gq_h}{2^k} + \frac{q_gq_s}{2^k} + \frac{q_g}{2^{\ell+1}} + \frac{q_f}{2^k} + \frac{q_d}{2^k}\right)\right),$$

avec $\tau' \leq \tau + (q_f + q_g + q_h + q_d)T_{lu} + q_d^2 T_{lu} + (q_d + 1)q_gq_h(T_\varphi + T_{lu})$, où T_φ est le temps pour une évaluation d'une fonction φ_{pk} et T_{lu}, le temps de recherche d'un élément dans une liste.

La preuve utilise une technique similaire à celle utilisée pour la sécurité sémantique d'OAEP 3-tours (voir la preuve du théorème 44). Elle considère les deux types d'attaque contre la signature et contre le chiffrement comme dans la preuve du théorème 47 dans le modèle de permutation aléatoire. Les détails sont dans [33].

6.4.3 Proposition de taille pour les paramètres

En utilisant les arguments similaires à ceux utilisées dans la section 6.3.3, on peut simplifier les contraintes sur les paramètres de sécurité :

– Pour le chiffrement, on a :

$$\frac{\varepsilon_E}{t} \leq \frac{\varepsilon'}{t} + \frac{Q}{2^k}.$$

Alors, $k = 161$ est convenable si les paramètres sont suffisamment grands, *i.e.* $\varepsilon'/t < 2^{-81}$).

– Pour la signature dans le cas général :

$$\frac{\varepsilon_S}{t} \leq \frac{Q\varepsilon'}{t} + \frac{Q}{2^k}.$$

Alors, $k = 161$ est aussi suffisant, à condition que $\varepsilon'/t < 2^{-161}$.

– Pour la signature dans le cas où la fonction φ_{pk} est induite par une permutation sans griffe :

$$\frac{\varepsilon_S}{t} \leq \frac{2\varepsilon'}{t} + \frac{Q}{2^k}.$$

On obtient une expression semblable à celle obtenue dans la construction dans la modèle de la permutation aléatoire pour le chiffrement : le terme ε'/t est remplacé par $2\varepsilon'/t$, ce qui permet des paramètres de sécurité plus courts. Par conséquent, $k = 161$ est suffisant, à condition que $\varepsilon'/t < 2^{-82}$.

En conclusion, pour le cas le plus intéressant de RSA, on peut choisir $k = 161$ à condition que le niveau de sécurité de la fonction φ soit de l'ordre de 2^{82}, *i.e.* un module de 1024 bits.

Diffusion de données chiffrées

7

Introduction

7.1 Traçage des traîtres

Le problème de sécurité dans la distribution de données est un des problèmes les plus importants de l'ère numérique. S'il n'y a pas assez de sécurité, on peut écrire des livres, composer de la musique mais on ne peut être payé qu'une seule fois car dès que le premier exemplaire est sorti, n'importe qui peut faire des copies et les distribuer. Les chaînes de télévision privées sont découragées car n'importe quel non - abonné peut capter ses programmes sans payer... Dans un tel monde, tous les fruits du travail sont des biens publics et les auteurs de ces travaux perdent tous leurs droits. Pour encourager l'investissement, il est indispensable que les problèmes de droit d'auteur, de sécurité, etc. occupent une place importante.

On peut distinguer deux façons de distribuer les données. La différence tient dans la nature du problème et l'objectif du pirate :

- Distribution « statique » de données. L'exemple type est la protection du contenu des CD-ROMs.
- Distribution « dynamique » de données. L'exemple type est la diffusion des programmes télévisés aux abonnés.

Concernant la distribution statique, un client, qui a acheté un CD-ROM par exemple peut toujours redistribuer le contenu de ce CD-ROM (on l'appellera alors « traîtres »). On ne peut pas l'en empêcher mais ce que l'on peut faire est de l'en décourager. La méthode utilisée est le « marquage » du produit, qui consiste à marquer quelques positions du contenu de telle manière que ces marques, bien que sans influence sur le contenu perçu, caractérisent chaque exemplaire. En un mot, chaque exemplaire est différent des autres, et si un client distribue le contenu de son exemplaire sans aucune modification, on peut le détecter. Le but ultime d'un traître ou d'une coalition de traîtres est de fabriquer un nouvel exemplaire et de le redistribuer sans être détectés. En revanche, l'objectif de l'autorité est de retrouver au moins un des traîtres de la coalition, source de l'exemplaire illégal.

Pour la distribution dynamique, où le contenu est transmis en temps réel sur un canal de diffusion, la technique de marquage est évidemment irréalisable car le centre ne peut émettre vers chaque abonné un programme différent. Le principe de la diffusion est d'envoyer un même signal à tous les abonnés. Pour y pallier, le contenu est chiffré avant d'être diffusé. Chaque abonné dispose donc d'un décodeur pour déchiffrer les signaux. Le but du pirate est de redistribuer le contenu. Une première méthode de fraude serait de redistribuer le contenu une fois déchiffré, mais cela suppose un système de diffusion. De plus, cette technique est facile à détecter et conduit souvent à des peines sévères. L'objectif réel du pirate est de construire lui-même un nouveau décodeur lui permettant de déchiffrer le signal émis par le centre. Pour ce faire, un traître, ou pire, une coalition de traîtres peut essayer d'ouvrir leur propre décodeur afin d'en extraire les clés de déchiffrement et en fabriquer un nouveau. Cependant, ce que ces traîtres veulent éviter (et que le centre veut atteindre) est qu'à partir d'un décodeur pirate, il soit possible de retrouver la source du piratage, *i.e.* un des traîtres.

Dans ce travail, nous nous concentrons sur le deuxième mode de distribution : le traçage de traîtres dans la distribution des données dynamiques. Il existe, dans la littérature, deux principales catégories de tels schémas :

1. Les schémas à taux de transmission *constant* [73]. Ces schémas conviennent particulièrement au chiffrement de messages volumineux. De plus, ils permettent une traçabilité efficace en *boîte noire*, *i.e.* la procédure de traçage n'a pas besoin d'ouvrir le décodeur pirate mais simplement d'interagir avec lui.
2. Les schémas à taille de textes clairs admissibles relativement petite [25, 22, 74]. Cependant, dans ce cas, le taux de transmission est *non constant* et est au moins linéaire en la taille de la coalition. De plus, pour ces schémas, il n'y pas de procédure de traçage en boîte noire efficace. Il est possible de faire du traçage en boîte noire [22] mais l'algorithme de traçage dans ce cas est irréaliste en raison de sa complexité plus grande que le coefficient binomial de n et c, où n est le nombre d'usagers et c est la taille maximale de la coalition.

Nous étudions ici la première catégorie en mettant l'accent sur un aspect important : l'optimisation du taux de transmission. Comme déjà mentionné, dans un système de transmission directe de données du centre aux abonnés, un point important est le rapport entre la taille du texte chiffré (le vrai signal envoyé par le centre) et celle du texte clair (le contenu du programme transmis aux abonnés). Dans ce cadre, les paramètres de transmission sont considérés : *le taux du texte chiffré*–le rapport entre la taille du texte chiffré et celle du texte clair ; *le taux de la clef de chiffrement*–le rapport entre la taille de la clef de chiffrement et celle du texte en clair dans un bloc primitif de données (car une clef est normalement utilisée plusieurs fois pour chiffrer plusieurs blocs de données) ; *le taux de la clef d'usager*–le rapport entre la taille de la clef d'usager et celle du texte clair dans un bloc primitif de données.

Plus intéressant, les systèmes de la première catégorie intègrent des codes résistants aux coalitions- la technique la plus connue pour faire du marquage proposée par Boneh-Shaw [25]. Cette méthode, proposée par Kiayias et Yung [73], va être utilisée dans notre construction. De plus, nous utilisons les propriétés particulières des couplages pour amé-

liorer l'efficacité et pour obtenir une nouvelle fonctionnalité : la traçabilité publique.

Avant de présenter notre construction, nous rappelons brièvement les applications des couplages en cryptographie. Puis, nous présentons notre attaque sur un schéma de traçage de traîtres fondé sur des couplages.

7.2 Couplages en cryptographie

La recherche sur les courbes elliptiques est une longue histoire, depuis les études des nombres congrues et des équations diophantiennes. Cependant, l'approche arithmétique pour étudier ces courbes est relativement nouvelle, et n'est vraiment utilisée que depuis le siècle dernier (avec les travaux de Mordell [85]) et la notion de groupes de points. Initialement, le champ de recherche dans cette direction était essentiellement limité aux courbes elliptiques sur des corps infinis avec le théorème fondamental selon lequel le groupe des points rationnels d'une courbe elliptique est engendré par un nombre fini de points. C'est dans les années 80 que les courbes elliptiques sur des corps finis ont trouvé, pour la première fois, leur application en cryptographie grâce aux travaux de Koblitz [75] et Miller [83]. Ces auteurs ont indépendamment trouvé que les schémas cryptographiques fondés sur la difficulté du problème du logarithme discret pourraient atteindre un meilleur niveau de sécurité avec un groupe de points sur une courbe elliptique qu'avec le groupe multiplicatif conventionnel d'un corps fini. En effet, à un même niveau de sécurité, l'utilisation des courbes elliptiques se contente d'une clef plus courte. À titre d'exemple, dans des applications cryptographiques, l'utilisation d'un groupe elliptique de taille environ 2^{160} garantit une sécurité équivalente à celle offerte par l'utilisation d'un groupe classique de taille 2^{1024}.

La raison de cet écart est que le problème du logarithme discret est en générale beaucoup plus difficile sur les courbes elliptiques. Ce n'est, cependant, pas toujours le cas. En 1993, Menezes, Okamoto et Vanstone [80] ont décrit la première attaque non triviale contre la difficulté du problème du logarithme discret sur certaines classes de courbes elliptiques. Ils ont prouvé que ce problème sur une courbe elliptique se réduit à ce même problème sur une extension du corps de base. Or, dans le cas de courbes elliptiques super singulières, le degré d'extension est très petit. Cette attaque, appelée la réduction MOV, utilise la propriété de bilinéarité du couplage de Weil. Un an après, Frey et Ruck [50] ont utilisé le couplage de Tate pour décrire une attaque similaire, appelée la réduction FR. Les couplages apportent des propriétés intéressantes : ce sont des applications bilinéaires qui sont efficacement calculables et non-dégénérées.

Ainsi les couplages furent-ils initialement considérés comme outil d'attaque et, pendant longtemps, ces applications cryptanalytiques étaient considérées comme les seules applications des couplages. C'est en 2000 qu'ils ont été utilisés pour la première fois comme un outil de construction de schémas cryptographiques : Joux [67] a construit un schéma d'échange de clef à trois parties qui généralise le protocole Diffie-Hellman de base ; Sakai *et. al* [110] ont ensuite proposé des schémas de signature et d'échange de clef fondés sur

l'identité. Ces travaux ouvrent un nouveau courant d'études : l'utilisation des couplages dans la construction des schémas cryptographiques. L'application la plus importante des couplages se trouve sans doute dans la construction de schémas fondés sur l'identité. En 1984, Shamir [112] a proposé une variante de la cryptographie à clef publique où l'on utilise l'identité comme la clef publique et la clef secrète correspondante est générée par un tiers de confiance. Le fait que la clef de chiffrement soit liée à l'identité de l'usager garantit implicitement son authenticité et exclut la nécessité d'avoir une infrastructure des clefs publiques pour la certification. La mise en œuvre d'un schéma de chiffrement efficace fondé sur l'identité restait cependant un problème ouvert jusqu'au travail de Boneh et Franklin [23]. Depuis ce résultat, plusieurs améliorations, aussi bien en termes d'efficacité qu'en termes de sécurité, ont été apportées [20, 21, 119]. Boneh *et. al* ont également utilisé les couplages pour construire des schémas de signature de petite taille [24, 26]. Ils ont èa cet effet utilisé des groupes où le problème CDH est difficile mais où le problème DDH est facile (pour la vérification). L'étude de tels groupes a été entreprise par Joux et Nguyen [69].

Aujourd'hui, cinq ans après la première utilisation des couplages, cet outil est largement utilisé dans presque toutes les branches de la cryptographie asymétrique. Le « Pairing-Based Crypto Lounge » de Barreto [3] regroupe environ deux cents articles. En ce qui concerne le problème de traçage de traîtres, le premier schéma fondé sur les couplages a été proposé par Mitsunari, Sakai et Kasahara [84] en 2002 sans fournir de preuve de sécurité. To, Safavi-Naini et Zhang [117] ont trouvé, en 2003, une attaque sur ce schéma et ont proposé de nouveaux schémas avec un nouvel algorithme de traçage de traîtres. Dans ce chapitre, nous montrons une attaque sur le schéma de To *et. al* et expliquons quelle était l'erreur dans leur analyse de sécurité. Dans le chapitre suivant, nous proposons une nouvelle construction qui devient la seule construction sûre fondée sur les couplages.

On rappelle brièvement la notion d'*application bilinéaire admissible*. On utilise les notations suivantes :

Notation 49

1. *$\mathcal{G}_1$ et $\mathcal{G}_2$ sont deux groupes cycliques d'ordre q, un grand nombre premier. $\mathcal{G}_1$ est un groupe additif et $\mathcal{G}_2$ est un groupe multiplicatif.*
2. *P est un générateur de $\mathcal{G}_1$.*
3. *e est une application $e : \mathcal{G}_1 \times \mathcal{G}_1 \to \mathcal{G}_2$.*
4. *$g = e(P, P)$.*

Définition 50 (Applications bilinéaires) *Étant donné deux groupes $\mathcal{G}_1$ et $\mathcal{G}_2$ (définis ci-dessus). Une application $e : \mathcal{G}_1 \times \mathcal{G}_1 \to \mathcal{G}_2$ est dite* application bilinéaire admissible *si elle satisfait les trois propriétés suivantes :*

Bilinéarité : $e(aU, bV) = e(U, V)^{ab}$ *quels que soient $U, V \in \mathcal{G}_1$ et $a, b \in \mathbb{Z}_q$;*

Non-dégénération : $e(P,P) \neq 1$;

Efficacement calculable : *Il existe un algorithme efficace pour calculer* $e(U,V)$ *pour tous* $U, V \in \mathcal{G}_1$.

Remarquons que, comme $\mathcal{G}_1, \mathcal{G}_2$ sont des groupes d'ordre premier et e est non-dégénérée, si P est un générateur de $\mathcal{G}_1$ alors $g = e(P,P)$ est un générateur de $\mathcal{G}_2$.

7.2.1 Exemples

On peut utiliser les couplages de Weil et de Tate pour construire des applications bilinéaires admissibles [118, 68].

7.2.2 Description du schéma de TSZ

Le schéma de To, Safavi-Naini et Zhang (appelé TSZ) est fondé sur une application bilinéaire $e : \mathcal{G}_1 \times \mathcal{G}_1 \to \mathcal{G}_2$, où $\mathcal{G}_1, \mathcal{G}_2$ sont des groupes d'ordre premier q.

Initialisation : choisir aléatoirement deux générateurs $P, Q \in \mathcal{G}_1$ et un polynôme unitaire de degré $2k-1$ à coefficients dans $\mathbb{Z}_q$:

$$f(x) = a_0 + a_1 x + \ldots + a_{2k-2} x^{2k-2} + x^{2k-1}.$$

Posons $Q_0 = a_0 Q$, $Q_1 = a_1 Q$, $\ldots$, $Q_{2k-2} = a_{2k-2} Q$ et $g = e(P,Q) \in \mathcal{G}_2$.

Clef secrète du centre : le générateur P et le polynôme f

Clef de chiffrement : le multiplet $(g, Q, Q_0, Q_1, \ldots, Q_{2k-2})$

Clef d'usager u **:** $K_u = f(u)^{-1} P$

Algorithme de chiffrement : l'algorithme de chiffrement génère un aléa $r \in \mathbb{Z}_q$. Ensuite, une clef de session $s \in \mathcal{G}_2$ est chiffrée par : $c = (sg^r, rQ, rQ_0, \ldots, rQ_{2k-2})$.

Algorithme de déchiffrement : grâce à la clef K_u, l'usager u calcule g^r pour retrouver s :

$$g^r = e(K_u, rQ_0) \cdot \ldots \cdot e(u^{2k-2} K_u, rQ_{2k-2}) \cdot e(u^{2k-1} K_u, rQ).$$

7.2.3 Attaque du schéma TSZ

Dans [117], les auteurs ont montré qu'une coalition d'au moins k usagers ne peut produire un décodeur anonyme. Ici, nous expliquons comment *un seul* usager peut construire un décodeur anonyme. Considérons un usager u, il choisit aléatoirement des éléments $z_0, z_1, \ldots, z_{2k-2} \in^R \mathbb{Z}_q$ et construit le décodeur suivant

– $X_i = u^i K_u + z_i Q$, pour i allant de 0 à $2k-2$.

– X_{2k-1} est déterminé par la relation :

$$X_{2k-1} = u^{2k-1}K_u - (z_0Q_0 + z_1Q_1 + \ldots + z_{2k-2}Q_{2k-2}). \tag{7.1}$$

L'usager u peut donc publier $X_0, X_1, \ldots, X_{2k-1}$, ce qui permet à n'importe qui de calculer g^r pour retrouver s :

$$g^r = e(X_0, rQ_0) \cdot \ldots \cdot e(X_{2k-2}, rQ_{2k-2}) \cdot e(X_{2k-1}, rQ).$$

Nous montrons que cette construction de décodeur est intraçable, *i.e.* à partir de $X_0, \ldots, X_{2k-2}, X_{2k-1}$, il est impossible de retrouver l'usager u. En effet, $X_0, \ldots, X_{2k-2}$, X_{2k-1} satisfont la relation suivante :

$$\begin{aligned} \sum_0^{2k-1} a_i X_i &= \left(\sum_0^{2k-2} a_i u^i\right) K_u + \left(\sum_0^{2k-2} a_i z_i\right) Q + u^{2k-1}K_u - \left(\sum_0^{2k-2} z_i a_i\right) Q \\ &= f(u)K_u = P. \end{aligned} \tag{7.2}$$

Remarquons aussi que, pour chaque usager et pour tout i ($i = 0, \ldots, 2k-2$), l'application de $\mathbb{Z}_q$ vers $\mathcal{G}_1$ qui envoie z_i sur X_i est une bijection. Alors, au lieu de choisir $z_0, z_1, \ldots, z_{2k-2}$, on peut choisir aléatoirement $X_0, \ldots, X_{2k-2}$ dans $\mathcal{G}_1$, qui détermine $(z_0, z_1, \ldots, z_{2k-2})$. La valeur X_{2k-1} est donc déterminée de façon unique par la relation (7.1). Elle satisfait également la relation (7.2). Comme elle détermine une valeur unique, on peut l'utiliser pour caractériser X_{2k-1} : elle est donc indépendante de l'usager u.

On peut facilement voir que tout usager peut produire un même ensemble de décodeurs pirates avec les paramètres $(X_0, X_1, \ldots, X_{2k-2}, X_{2k-1})$ où $X_0, X_1, \ldots, X_{2k-2}$ sont aléatoirement choisis dans $\mathcal{G}_1^{2k-1}$ et X_{2k-1} est défini par la relation (7.2).

7.2.4 Faille dans l'analyse de sécurité

Notons que cette attaque est bien différente de celle contre le schéma de Mitsunari *et. al* dans [117]. Notre construction de décodeur pirate combine en effet l'information de la clef privée d'un usager et l'*information publique* du système. Elle montre une faille dans l'algorithme de traçage dans [117].

La procédure de traçage dans [117] fonctionne comme suit : pour un ensemble d'usagers suspects $A = \{u_1, \ldots, u_t\}$ ($t \leq k$), on construit un autre polynôme $f'(x) = f(x) + \alpha(x - u_1)\ldots(x - u_t)$. Pour chaque usager dans A, la clef de u est la même qu'elle soit dans le schéma utilisant f (appelé $\mathsf{S}(f)$) ou dans le schéma utilisant f' (appelé $\mathsf{S}(f')$). De ce fait, To *et. al* déclarent que si tous les traîtres sont dans A, alors tout décodeur pirate produit par ces traîtres dans $\mathsf{S}(f)$ est aussi un décodeur pirate dans $\mathsf{S}(f')$. Par conséquent, ce décodeur pirate déchiffre de la même façon un texte chiffré selon $\mathsf{S}(f')$ et un texte chiffré selon $\mathsf{S}(f)$. Grâce à cet argument et à l'envoi d'un texte chiffré au décodeur pirate, le centre peut facilement détecter si les traîtres sont inclus dans A.

Malheureusement, ce raisonnement est fausse. En effet, il suppose que la construction du décodeur pirate ne dépende que des clefs privées des usagers et pas des informations

publiques. Lorsque la construction du décodeur dépend aussi des informations publiques (qui sont évidemment disponibles aux traîtres), l'algorithme de traçage du schéma échoue, comme le montre le contre-exemple ci-dessus.

8

Traçabilité publique dans le traçage de traîtres

Sommaire

À Eurocrypt '02, Kiayias et Yung [73] ont proposé un schéma de traçage de traîtres à taux de transmission constant (noté KY dans la suite) : le taux de texte chiffré (le ratio entre la taille du chiffré et celle du clair) est 3, le taux de la clef de chiffrement (le ratio entre la taille de la clef de chiffrement et celle du clair) est 4 et le taux de la clef de l'usager (le ratio entre la taille de la clef d'usager et celle du clair) est 2.

Dans ce chapitre, nous apportons quelques propositions qui améliorent ces taux : le taux de texte chiffré est réduit au taux optimal de 1 (asymptotiquement) ; le taux de la clef de chiffrement est réduit à 1 et le taux de la clef de l'usager reste inchangé. Remarquons

que ces taux de transmission sont considérés dans le cas de plusieurs usagers, lorsque le nombre d'usagers et la taille du texte clair sont grands. Ces améliorations sont obtenues tout en préservant deux propriétés extrêmement précieuses pour un schéma de traçage de traîtres, comme dans le schéma KY :

- diffusion de données chiffrées à *clef publique*, où une tierce personne est capable d'envoyer des messages confidentiels aux abonnés ;
- traçage de traîtres efficace en *boîte noire* où la procédure de traçage peut s'effectuer sans ouvrir le décodeur pirate.

De plus, nous proposons une nouvelle fonctionnalité intéressante : la *traçabilité publique*. Dans les schémas précédents, on a besoin d'informations privées (importantes) pour retrouver les traîtres. Dans notre schéma, le centre peut publier des informations de telle sorte que n'importe qui est capable d'exécuter la procédure de traçage, au moins dans la phase la plus coûteuse d'interaction avec le décodeur pirate.

8.1 Hypothèses calculatoires

Dans ce qui suit, on utilise le terme « couplage » pour signifier les couplages modifiés de Weil ou de Tate qui satisfont les propriétés d'une application bilinéaire admissible. Dans cette section, on considère des groupes $\mathcal{G}_1$ et $\mathcal{G}_2$ sur lesquels il existe un couplage $e : \mathcal{G}_1 \times \mathcal{G}_1 \to \mathcal{G}_2$. .

8.1.1 Hypothèses classiques et variantes

Nous rappelons quelques problèmes bien connus dans les groupes dotés d'un couplage , tels que le problème calculatoire bilinéaire de Diffie-Hellman. Nous proposons également quelques nouveaux problèmes.

Tout d'abord, on rappelle quelques problèmes classiques dans le groupe $\mathcal{G}_1$.

CDH – le problème calculatoire Diffie-Hellman dans $\mathcal{G}_1$:
Étant donné (P, aP, bP), où $a, b \overset{R}{\leftarrow} \mathbb{Z}_q^\star$, calculer abP.

CBDH^1 – le problème calculatoire Diffie-Hellman bilinéaire dans $\mathcal{G}_1$:
Étant donné (P, aP, bP, cP), où $a, b, c \overset{R}{\leftarrow} \mathbb{Z}_q^\star$, calculer $abcP$.

DBDH^1 – le problème décisionnel Diffie-Hellman bilinéaire dans $\mathcal{G}_1$:
Étant donné (P, aP, bP, cP, U), où $a, b, c \overset{R}{\leftarrow} \mathbb{Z}_q^\star$ et $U \overset{R}{\leftarrow} \mathcal{G}_1$, décider si $U = abcP$.

Nous introduisons maintenant deux versions modifiées des deux problèmes bilinéaires Diffie-Hellman. Elles sont, en effet, des cas particuliers, où $b = c$. Nous prouvons donc quelques relations entre ces problèmes et le problème usuel CDH .

$\mathsf{CBDH}^1\text{–}\mathsf{M}$ – le problème calculatoire Diffie-Hellman bilinéaire modifié dans $\mathcal{G}_1$:
Étant donné (P, aP, bP), où $a, b \overset{R}{\leftarrow} \mathbb{Z}_q^\star$, calculer ab^2P.

$\mathsf{DBDH}^1-\mathsf{M}$ – le problème décisionnel Diffie-Hellman bilinéaire modifié dans $\mathcal{G}_1$:
Étant donné (P, aP, bP, U), où $a, b \stackrel{R}{\leftarrow} \mathbb{Z}_q^\star$ et $U \stackrel{R}{\leftarrow} \mathcal{G}_1$, décider si $U = ab^2P$.

Proposition 51 *Le problème* $\mathsf{CBDH}^1-\mathsf{M}$ *est au moins aussi difficile que le problème* CBDH^1*, qui est au moins aussi difficile que le problème* CDH *:*

$$(\mathsf{Succ}_{\mathcal{G}_1}^{\mathsf{CBDH}^1-\mathsf{M}}(t))^2 \leq \mathsf{Succ}_{\mathcal{G}_1}^{\mathsf{CBDH}^1}(2t + 3T_e) \leq \mathsf{Succ}_{\mathcal{G}_1}^{\mathsf{CDH}}(2t + 4T_e),$$

où T_e *est le temps de calcul d'une exponentielle dans le groupe* $\mathcal{G}_1$*.*

Preuve. On montre d'abord l'inégalité de gauche. Considérons un attaquant $\mathcal{A}$ qui peut résoudre le problème $\mathsf{CBDH}^1-\mathsf{M}$ avec au moins une probabilité de succès ε. On peut alors construire un algorithme $\mathcal{B}$ qui résout le problème CBDH^1 avec au moins une probabilité de succès ε^2. Une instance aléatoire $(P, A = aP, B = bP))$ du problème CBDH^1 est donnée comme entrée à $\mathcal{B}$. $\mathcal{B}$ trouve la solution $D = abcP$ en interagissant avec $\mathcal{A}$ de la manière suivante :

Étape 1 : $\mathcal{B}$ calcule $X_1 = A - B = (a - b)P$ et envoie αC et X_1 à $\mathcal{A}$ où α est un aléa $\alpha \stackrel{R}{\leftarrow} \mathbb{Z}_q^\star$. $\mathcal{A}$ retourne donc $Y_1 = \alpha c(a - b)^2P$ avec probabilité ε.

Étape 2 : $\mathcal{B}$ calcule $X_2 = A + B = (a + b)P$ et envoie βC et X_2 à $\mathcal{A}$ où β est un aléa $\beta \stackrel{R}{\leftarrow} \mathbb{Z}_q^\star$. $\mathcal{A}$ retourne donc $Y_2 = \beta c(a + b)^2P$ avec probabilité ε.

Étape 3 : L'algorithme $\mathcal{B}$ retourne

$$\begin{aligned} D &= (4^{-1} \bmod q)(\beta^{-1}Y_2 - \alpha^{-1}Y_1)P \\ &= 4^{-1}c\left((a + b)^2 - (a - b)^2\right)P = 4^{-1}c(4ab)P \\ &= abcP. \end{aligned}$$

Comme a et b sont deux aléas indépendants, $a + b$ et $a - b$ le sont aussi. Par conséquent, $\mathcal{B}$ retourne une solution correcte avec une probabilité plus grande que ε^2.

On démontre maintenant l'inégalité de droite.

Considérons un attaquant $\mathcal{A}$ qui peut résoudre le problème CBDH^1 avec une probabilité de succès ε. On peut alors construire un algorithme $\mathcal{B}$ qui résout le problème CDH avec une probabilité de succès ε. Une instance aléatoire $(P, A = aP, B = bP, C = cP))$ du problème CDH est donnée comme entrée à $\mathcal{B}$. $\mathcal{B}$ trouve la solution $D = abP$ en interagissant avec $\mathcal{A}$, de la manière suivante :

Étape 1 : $\mathcal{B}$ choisit aléatoirement $c \stackrel{R}{\leftarrow} \mathbb{Z}_q$ et envoie $(P, A, B, C = cP)$ à $\mathcal{A}$. $\mathcal{A}$ retourne donc $U = abcP$ avec probabilité ε.

Étape 2 : $\mathcal{B}$ retourne $D = c^{-1}U$.

On voit facilement que $\mathcal{B}$ retourne une solution correcte avec une probabilité plus grande que ε. □

8.1.2 Nouveaux problèmes fondés sur les couplages

Nous rappelons maintenant les problèmes Diffie-Hellman bilinéaires dans les groupes $\mathcal{G}_1$ et $\mathcal{G}_2$.

CBDH^2 – Le problème calculatoire Diffie-Hellman bilinéaire :
Étant donné (P, aP, bP, cP), où $a, b, c \stackrel{R}{\leftarrow} \mathbb{Z}_q^\star$.
Calculer g^{abc}.

DBDH^2 – Le problème décisionnel Diffie-Hellman bilinéaire :
Étant donné (P, aP, bP, cP, Z), où $a, b, c \stackrel{R}{\leftarrow} \mathbb{Z}_q^\star$ et $Z \stackrel{R}{\leftarrow} \mathcal{G}_2$. Décider si $Z = g^{abc}$.

$\mathsf{CBDH}^2{-}\mathsf{E}$ – Le problème calculatoire Diffie-Hellman bilinéaire étendu :
Étant donné (P, aP, bP, cP, ab^2P), où $a, b, c \stackrel{R}{\leftarrow} \mathbb{Z}_q^\star$. Calculer g^{cb^2}.

$\mathsf{DBDH}^2{-}\mathsf{E}$ – Le problème décisionnel Diffie-Hellman bilinéaire étendu :
Étant donné $(P, aP, bP, cP, ab^2P, Z)$, où $a, b, c \stackrel{R}{\leftarrow} \mathbb{Z}_q^\star$ et $Z \stackrel{R}{\leftarrow} \mathcal{G}_2$.
Décider si $Z = g^{cb^2}$.

Nous introduisons également une variante du problème CBDH^2 avec le but de conforter la difficulté du problème de $\mathsf{CBDH}^2{-}\mathsf{E}$

$\mathsf{CBDH}^2{-}\mathsf{V}$ – Une variante du problème calculatoire Diffie-Hellman bilinéaire :
Étant donné $(P, aP, bP, cP, a(a^2 - b^2)P, b(a^2 - b^2)P)$, où $a, b, c \stackrel{R}{\leftarrow} \mathbb{Z}_q^\star$.
Calculer g^{abc}.

Proposition 52 *Le problème* $\mathsf{CBDH}^2{-}\mathsf{E}$ *est au moins aussi difficile que le problème* $\mathsf{CBDH}^2{-}\mathsf{V}$ *:*

$$(\mathsf{Succ}^{\mathsf{CBDH}^2-\mathsf{E}}_{e,\mathcal{G}_1,\mathcal{G}_2}(t))^2 \leq \mathsf{Succ}^{\mathsf{CBDH}^2-\mathsf{V}}_{e,\mathcal{G}_1,\mathcal{G}_2}(t + T_e),$$

où T_e est le temps de calcul d'une exponentielle dans le groupe $\mathcal{G}_2$.

Preuve. Considérons un attaquant $\mathcal{A}$ qui peut résoudre le problème $\mathsf{CBDH}^2{-}\mathsf{E}$ avec une probabilité de succès ε. On peut alors construire un algorithme $\mathcal{B}$ qui résout le problème $\mathsf{CBDH}^2{-}\mathsf{V}$ avec une probabilité de succès ε^2. Une instance aléatoire $(P, A = aP, B = bP, C = cP, U = a(a^2-b^2)P, V = b(a^2-b^2)P)$ du problème $\mathsf{CBDH}^2{-}\mathsf{V}$ est donnée comme entrée à $\mathcal{B}$. $\mathcal{B}$ trouve la solution $Z = g^{abc}$, en interagissant avec $\mathcal{A}$, de la manière suivante :

Étape 1 : $\mathcal{B}$ calcule $xP = A + B = (a + b)P$, $yP = A - B = (a - b)P$, $zP = cP$ et $xy^2P = U - V$. $\mathcal{B}$ envoie (xP, yP, zP, xy^2P) à $\mathcal{A}$. $\mathcal{A}$ retourne $Z_1 = g^{zy^2} = g^{c(a-b)^2}$ avec probabilité ε.

Étape 2 : $\mathcal{B}$ calcule $yx^2P = U + V$ et envoie (yP, xP, zP, yx^2P) à $\mathcal{A}$. $\mathcal{A}$ retourne $Z_2 = g^{zx^2} = g^{c(a+b)^2}$ avec probabilité ε.

Étape 3 : $\mathcal{B}$ retourne $Z = (Z_2/Z_1)^{4^{-1} \bmod q}$.

Du fait que x, y, z sont des aléas indépendants, on peut facilement rendre aléatoires les instances envoyées à $\mathcal{A}$. On voit donc que $\mathcal{B}$ retourne une solution correcte avec probabilité ε^2. □

Problèmes mixtes

Nous introduisons quelques nouveaux problèmes qui concernent des éléments aussi bien de $\mathcal{G}_1$ que de $\mathcal{G}_2$.

MCDH – le problème calculatoire Diffie-Hellman mixte :
Étant donné (P, aP, a^2P, g^b), où $a, b \overset{R}{\leftarrow} \mathbb{Z}_q^\star$, et $g = e(P, P)$.
Calculer g^{ba^2}.

MDDH – le problème décisionnel Diffie-Hellman mixte :
Étant donné (P, aP, a^2P, g^b, Z), où $a, b \overset{R}{\leftarrow} \mathbb{Z}_q^\star$, $Z \overset{R}{\leftarrow} \mathcal{G}_2$, Et $g = e(P, P)$.
Décider si $Z = g^{ba^2}$.

Convention 53 *Pour un problème X, on appelle « l'hypothèse X » l'hypothèse selon laquelle X est difficile à résoudre. Autrement dit, il n'existe pas d'algorithme polynomial probabiliste qui peut résoudre X avec une probabilité non négligeable.*

8.2 Schéma à deux usagers

8.2.1 Discussion sur l'utilisation des hypothèses

Nous avons introduit plusieurs nouveaux problèmes qui servent aux analyses de sécurité. Avant d'étudier ces problèmes de sécurité, résumons d'abord notre choix d'utilisation des hypothèses selon le niveau de sécurité.

Traçage de traîtres

Considérons d'abord la sécurité sémantique du schéma de chiffrement, *i.e.* la confidentialité vis-à-vis des non-abonnée. Dans le modèle de l'oracle aléatoire, la sécurité est assurée sous l'hypothèse MCDH. Dans le modèle standard, la sécurité est assurée sous une hypothèse plus forte : l'hypothèse MDDH.

En ce qui concerne le traçage de traîtres, on considère d'abord une propriété plus faible : la *non-incrimination*, *i.e.* un traître ne peut pas incriminer un innocent. On montre que, sous l'hypothèse CDH, notre schéma atteint cette propriété. La propriété de traçabilité de traîtres, *i.e.* dénoncer un véritable traître sans se tromper, est assurée sous l'hypothèse $\mathsf{DBDH}^1{-}\mathsf{M}$.

En un mot, notre schéma atteint les niveaux de sécurité classiques de traçage de traîtres sous les hypothèses MDDH et $\mathsf{DBDH}^1{-}\mathsf{M}$.

Traçabilité publique

Notre schéma apporte la nouvelle propriété intéressante de *traçabilité publique*. Cependant, pour l'obtenir, nous avons besoin d'hypothèses plus fortes car l'attaquant dispose de beaucoup plus d'informations.

Pour la sécurité sémantique du schéma de chiffrement, dans le modèle de l'oracle aléatoire, l'hypothèse $\mathsf{CBDH}^2-\mathsf{E}$, une petite extension de l'hypothèse classique CBDH^2 (voir la proposition 52), est exigée. Dans le modèle standard, la sécurité est fondée sur l'hypothèse $\mathsf{DBDH}^2-\mathsf{E}$.

Au sujet du traçage de traîtres, le traçage d'un traître est assurée sous l'hypothèse $\mathsf{DBDH}^1-\mathsf{M}$ (ce qui implique la non-incrimination).

Conclusion

Notre schéma apporte la propriété de traçabilité publique fondée essentiellement sur trois nouvelles hypothèses : MDDH pour la sécurité du chiffrement, $\mathsf{DBDH}^1-\mathsf{M}$ pour la propriété de traçage de traître, et $\mathsf{DBDH}^2-\mathsf{E}$ pour la traçabilité publique en boîte noire.

8.2.2 Schéma de Kiayias-Yung

Notre construction est inspirée du schéma de Kiayias et Yung [73], qui est, à son tour, considéré comme un cas particulier du schéma de Boneh et Franklin [22]. Nous allons modifier le schéma KY pour rendre possible l'utilisation des couplages.

Tout d'abord, rappelons le schéma KY.

Implémentation : Étant donné un paramètre de sécurité $\kappa \in \mathbb{Z}$, le schéma de traçage de traîtres fonctionne de la manière suivante :

Étape 1 : Générer un nombre premier q de κ bits et un groupe $\mathcal{G}$ d'ordre q. Choisir aléatoirement un générateur $g \in \mathcal{G}$.

Étape 2 : Prendre des éléments aléatoires $a, z \overset{R}{\leftarrow} \mathbb{Z}_q^\star$, et poser $Q = g^a$, $Z = g^z$.

Clef secrète du centre : le couple (a, z)

Clef de chiffrement : $\mathsf{pk} = (g, Q, Z)$

Clef d'usager : L'usager u_b (pour $b \in \{0, 1\}$, car on se focalise dans un premier temps sur le cas à deux usagers), est associé à une « représentation » $k_b = (\alpha_b, \beta_b)$ de g^z dans la base (g, g^a), *i.e.* l'autorité choisit deux vecteurs (α_0, β_0) et (α_1, β_1) dans $\mathbb{Z}_q^2$ tels que $\alpha_b + a\beta_b = z \bmod q$ pour tout $b \in \{0, 1\}$. Ces deux vecteurs sont choisis pour être linéairement indépendants. L'ensemble des clefs possibles est :

$$\mathcal{K}_{\mathsf{pk}} = \{(\alpha, \beta) | \alpha + a\beta = z \bmod q\}.$$

Algorithme de chiffrement : L'algorithme de chiffrement génère un aléa $k \in \mathbb{Z}_q$ et retourne un texte chiffré (c_1, c_2, d) dans $\mathcal{G}^3$: on a le texte clair m, supposé être dans le groupe $\mathcal{G}$, le centre calcule $C = (c_1 = g^k, c_2 = Q^k, d = m \cdot Z^k)$. On dit que le triplet $(c_1, c_2, d) \in \mathcal{G}^3$ est un texte chiffré valide s'il existe $k \in \mathbb{Z}_q$ tel que $c_1 = g^k$ et $c_2 = Q^k$. Sinon, le texte chiffré est invalide.

Algorithme de déchiffrement : Sur le texte chiffré (c_1, c_2, d), l'usager u_b calcule :

$$Z^k = c_1^\alpha \cdot c_2^\beta \quad \text{et} \quad m = d/Z^k.$$

8.2.3 Notre construction

Dans cette section, nous montrons comment on peut utiliser les applications bilinéaires pour améliorer ce schéma. Plus précisément, nous introduisons la notion de traçage de traîtres à clef publique avec « *sous-clef de déchiffrement équivalente* » : l'autorité génère pour chaque usager une clef et une *sous-clef de déchiffrement équivalente* qui lui correspond. L'autorité conserve la clef d'usager et ne donne à l'usager que la seule sous-clef (de déchiffrement équivalente) qui est suffisante pour le déchiffrement. La clef d'usager sera ultérieurement utilisée pour le traçage.

Implémentation : Étant donné un paramètre de sécurité $\kappa \in \mathbb{Z}$, le schéma de traçage de traîtres fonctionne de la façon suivante :

Étape 1 : Générer un nombre premier q de κ bits et deux groupes $\mathcal{G}_1$ et $\mathcal{G}_2$ d'ordre q ainsi qu'une application bilinéaire admissible $e : \mathcal{G}_1 \times \mathcal{G}_1 \to \mathcal{G}_2$. Choisir aléatoirement un générateur $P \xleftarrow{R} \mathcal{G}_1$ et poser $g = e(P, P)$, un générateur du groupe $\mathcal{G}_2$.

Étape 2 : Choisir des éléments aléatoires $a, z \xleftarrow{R} \mathbb{Z}_q^\star$, et poser $Q = aP$, $Z = g^z$.

Étape 3 : Choisir une fonction $H : \mathcal{G}_1 \to \mathcal{M}$. L'analyse de sécurité considère H comme un oracle aléatoire ou une fonction de hachage universelle (avec le « leftover-hash-lemma » [64, 65]).

L'espace des messages est $\mathcal{M} = \{0,1\}^{\kappa'}$ (la taille κ' du masque dépend du modèle utilisé : fonction de hachage universelle ou oracle aléatoire). L'espace des textes chiffrés est $\mathcal{G}_1^\star \times \mathcal{G}_1^\star \times \{0,1\}^{\kappa'}$. Les paramètres du système sont $\mathsf{params} = (q, \mathcal{G}_1, \mathcal{G}_2, e, P, H)$.

Clef secrète du centre : le couple (a, z).

Clef de chiffrement : le multiplet $\mathsf{pk} = (g, Q, Z)$.

Clef d'usager : l'usager u_b (pour $b \in \{0,1\}$) est associé à une « représentation » $k_b = (\alpha_b, \beta_b)$ de g^z dans la base (g, g^a). L'ensemble des clefs possibles est :

$$\mathcal{K}_{\mathsf{pk}} = \{(\alpha, \beta) | \alpha + a\beta = z \bmod q\}$$

Remarquons que l'autorité génère ces clefs pour les usagers mais il ne les leur donne pas. Seule une sous-clef de déchiffrement équivalente est fournie à chaque usager. Cette sous-clef équivalente est décrite ci-dessous.

Sous-clef de déchiffrement équivalente : l'usager u_b (pour $b \in \{0,1\}$) reçoit une sous-clef de déchiffrement équivalente $\Pi(k_b) = (\alpha_b, \pi_b = \beta_b P)$. L'ensemble de ces clefs est :

$$\Pi_{\mathsf{pk}} = \{(\alpha, \pi = \beta P) | (\alpha, \beta) \in \mathcal{K}_{\mathsf{pk}}\}.$$

Algorithme de chiffrement : L'algorithme de chiffrement génère un aléa $k \in \mathbb{Z}_q$ et retourne un texte chiffré (c_1, c_2, d) dans $\mathcal{G}_1 \times \mathcal{G}_1 \times \mathcal{M}$: sur le texte clair $m \in \mathcal{M}$; le centre calcule $C = (c_1 = kP, c_2 = k^2 Q, d = m \oplus H(Z^{k^2}))$. On dit que $(c_1, c_2, d) \in \mathcal{G}_1 \times \mathcal{G}_1 \times \mathcal{M}$ est un texte chiffré valide s'il existe $k \in \mathbb{Z}_q$ tel que $c_1 = kP$ et $c_2 = k^2 Q$. Sinon, il est dit invalide.

Algorithme de déchiffrement : Sur un texte chiffré (c_1, c_2, d), l'usager u_b calcule, grâce à sa sous-clef de déchiffrement équivalente $\Pi(k_b) = (\alpha_b, \pi_b)$,

$$Z^{k^2} = e(\alpha_b c_1, c_1) \cdot e(\pi_b, c_2) \quad \text{et} \quad m = d \oplus H(Z^{k^2}).$$

8.2.4 Raisonnement

Tout d'abord, il est naturel de se poser la question : « Peut-on chiffrer le message par $c_1 = kP$, $c_2 = kQ$ et $d = m \oplus H(Z^k)$, et donner à chaque usager une clef $k_b = (\alpha_b, \beta_b)$ (telle que $\alpha_b + a\beta_b = z \bmod q$) ? ». Ce schéma serait plus simple et serait, en effet, une transformation directe du schéma KY en utilisant les couplages. Cependant, de la même façon que l'attaque contre le schéma TSZ, présentée dans le chapitre précédent, l'attaquant pourrait profiter des propriétés des couplages pour casser le schéma. En effet, même s'il ne peut pas produire une nouvelle clef, il peut produire et distribuer un décodeur anonyme : $(X = \alpha P - uQ, Y = \beta P + uP)$, où u est aléatoirement choisi dans $\mathbb{Z}_q$. Alors, n'importe qui peut utiliser ce décodeur pour retrouver $Z^k = e(X, c_1) \cdot e(Y, c_2)$ et comme u est aléatoirement choisi, l'autorité ne peut pas tracer le traître.

Dans notre schéma, nous prouvons l'impossibilité d'un tel attaquant : l'usager ne dispose pas de clef de la forme (α, β), mais seulement une sous-clef de déchiffrement équivalente de forme $(\alpha, \beta P)$. Ceci entraîne que même si deux usagers s'entendent pour tricher, ils ne peuvent rien apprendre de la clef secrète (a, z). On va voir que cette propriété est primordiale pour améliorer le résultat dans le cas de plusieurs usagers.

8.2.5 Sécurité du schéma de chiffrement

Avant de considérer les propriétés spécifiques du traçage de traîtres, nous étudions la sécurité du schéma de chiffrement. Nous montrons que si on considère la fonction H comme un oracle aléatoire, la sécurité sémantique du chiffrement sera fondée sur l'hypothèse MCDH et, si on considère la fonction H comme aléatoirement choisie dans une famille de fonctions de hachage universelles [64, 65], la sécurité sémantique du chiffrement sera fondée sur l'hypothèse MDDH.

Théorème 54 *Considérons H comme un oracle aléatoire. Le schéma de chiffrement est sémantiquement sûr sous l'hypothèse* MCDH.

Preuve. Considérons un attaquant $\mathcal{A}$ contre le schéma de chiffrement, selon une attaque IND-CPA. Supposons que $\mathcal{A}$ peut avoir un avantage ε, alors on peut résoudre le problème MCDH avec une probabilité de succès ε/q_H, où q_H est le nombre de requêtes que $\mathcal{A}$ soumet à l'oracle H. En effet, on va construire un algorithme $\mathcal{B}$ qui, en utilisant $\mathcal{A}$ comme un sous-programme, résout le problème MCDH.

Une instance aléatoire du problème MCDH $(P, A = kP, B = k^2P, C = g^z))$ est donnée à $\mathcal{B}$ et $\mathcal{B}$ en trouve la solution $D = g^{zk^2}$ de la manière suivante :

Implémentation : $\mathcal{B}$ choisit aléatoirement a, définit la clef publique $\mathsf{pk} = (g, Q = aP, Z = C)$ et transmet pk à $\mathcal{A}$.

Requête-H : $\mathcal{B}$ maintient une liste $\mathcal{H}$-List pour répondre aux requêtes de l'oracle H à l'attaquant $\mathcal{A}$.

Challengeur : Quand $\mathcal{A}$ retourne deux candidats m_0, m_1, $\mathcal{B}$ prend un élément aléatoire $d \in \mathcal{M}$ et transmet (A, aB, d) comme challenge à $\mathcal{A}$. Ceci est une simulation parfaite sauf si g^{zk^2} a déjà été demandé à l'oracle H. Dans cette simulation, $\mathcal{A}$ a un avantage nul car d est indépendant de m_0, m_1.

Choix : Quand l'attaquant $\mathcal{A}$ devine $b' \in \{0, 1\}$, l'algorithme $\mathcal{B}$ prend aléatoirement (X, h) dans $\mathcal{H}$-List et retourne X.

On voit facilement que l'algorithme $\mathcal{B}$ retourne une solution correcte $X = D$ avec probabilité ε/q_H : la seule façon pour $\mathcal{A}$ d'avoir un certain avantage sur le schéma est de demander D à H. □

Théorème 55 *Considérons H une fonction aléatoirement choisie dans une famille de fonctions de hachage universelles. Le schéma est sémantiquement sûr sous l'hypothèse* MDDH.

Preuve. Considérons un attaquant $\mathcal{A}$ contre le schéma de chiffrement, selon une attaque IND-CPA. Supposons que $\mathcal{A}$ peut avoir un avantage ε, alors on peut résoudre le problème MDDH avec probabilité $\varepsilon/2$. En effet, on va construire un algorithme $\mathcal{B}$ qui, en utilisant $\mathcal{A}$ comme un sous-programme, résout le problème MDDH.

Une instance aléatoire du problème MDDH $(P, A = kP, B = k^2P, C = g^z, U))$ où U, qui est soit la solution du problème MCDH soit aléatoirement choisi dans $\mathcal{G}_2$, est donnée à $\mathcal{B}$. $\mathcal{B}$ distingue ces deux cas de la façon suivante :

Implémentation : $\mathcal{B}$ choisit aléatoirement a, pose la clef publique $\mathsf{pk} = (g, Q = aP, Z = C)$ et transmet pk à $\mathcal{A}$.

Challenger : Quand $\mathcal{A}$ retourne deux candidats m_0, m_1, $\mathcal{B}$ prend un bit aléatoire $b \in \{0, 1\}$ et transmet $(A, aB, d = m_b \oplus H(U))$ comme challenge à $\mathcal{A}$.

Choix : Quand l'attaquant $\mathcal{A}$ devine $b' \in \{0,1\}$, l'algorithme $\mathcal{B}$ retourne 1 (*i.e.* U est la solution au problème MCDH) si $b = b'$ et 0 sinon.

On observe que si U est la solution au problème MCDH, alors le texte chiffré C est un chiffrement de m_b. Sinon, comme H est aléatoirement choisie dans une famille de fonctions de hachage universelles, C est le chiffrement d'un message aléatoire, b est donc égal à b' avec probabilité 1/2. Par un argument classique, on trouve que l'attaquant $\mathcal{B}$ dispose d'un avantage de $\varepsilon/2$ pour résoudre le problème MDDH. □

8.2.6 Non-incrimination

Dans plusieurs situations, on se contente de la propriété de non-incrimination, *i.e.* un traître, en utilisant les informations de sa sous-clef de déchiffrement équivalente, ne peut fabriquer un décodeur pirate correspondant à la sous-clef d'un usager innocent. Évidemment, cette propriété est plus faible que la propriété de traçage de traîtres.

Théorème 56 *Étant donné la clef de chiffrement et une sous-clef de déchiffrement équivalente* $(\alpha, \pi) \in \Pi_{\mathsf{pk}}$*, il est calculatoirement difficile de fabriquer une autre sous-clef de déchiffrement équivalente dans* Π_{pk} *sous l'hypothèse* CDH*.*

Preuve. Considérons un attaquant $\mathcal{A}$, un traître, dont le but est d'incriminer un usager innocent. Étant donné une sous-clef de déchiffrement équivalente $(\alpha, \pi) \in \Pi_{\mathsf{pk}}$, supposons que $\mathcal{A}$ puisse construire une autre sous-clef de déchiffrement équivalente. Alors, on peut résoudre le problème inverse de Diffie-Hellman (*i.e.* à partir de (P, aP), calculer $a^{-1}P$) qui est connu pour être équivalent au problème classique Diffie-Hellman CDH. En effet, on va construire un algorithme $\mathcal{B}$ qui, en utilisant $\mathcal{A}$ comme un sous-programme, résout le problème inverse de CDH.

Une instance aléatoire du problème inverse de CDH $(P, A = aP)$ est donnée à $\mathcal{B}$ et $\mathcal{B}$ en trouve la solution $B = a^{-1}P$ de la manière suivante :

Implémentation : $\mathcal{B}$ choisit aléatoirement $\alpha \overset{R}{\leftarrow} \mathbb{Z}_q^\star$ et $\pi \overset{R}{\leftarrow} \mathcal{G}_1$, puis calcule $Z = e(P, \alpha P)e(A, \pi)$. Finalement, $\mathcal{B}$ définit la clef publique $\mathsf{pk} = (g, A, Z)$ et transmet pk ainsi que la sous-clef de déchiffrement équivalente (α, π) à $\mathcal{A}$.

Attaque : $\mathcal{A}$ retourne une autre sous-clef de déchiffrement équivalente $(\tilde{\alpha}, \tilde{\pi})$.

Solution : $\mathcal{B}$ calcule $c = \tilde{\alpha} - \alpha$ et retourne $B = (c^{-1} \bmod q)(\pi - \tilde{\pi})$.

Puisque $(\tilde{\alpha}, \tilde{\pi})$ est une nouvelle sous-clef de déchiffrement équivalente, pour tout texte chiffré (c_1, c_2, d),

$$\begin{aligned} e(\alpha c_1, c_1) \cdot e(\pi, c_2) &= e(\tilde{\alpha} c_1, c_1) \cdot e(\tilde{\pi}, c_2) \\ \iff e(\alpha k P, kP) \cdot e(\pi, ak^2 P) &= e(\tilde{\alpha} k P, kP) \cdot e(\tilde{\pi}, ak^2 P) \\ \iff e(\alpha P, P) \cdot e(\pi, aP) &= e(\tilde{\alpha} P, P) \cdot e(\tilde{\pi}, aP) \\ \iff e((\tilde{\alpha} - \alpha)P, P) &= e(\pi - \tilde{\pi}, aP). \end{aligned}$$

Du fait que $(\alpha, \pi), (\tilde{\alpha}, \tilde{\pi}) \in \Pi_{\mathsf{pk}}$, on voit que π et $\tilde{\pi}$ sont dans le groupe $\mathcal{G}_1$ généré par P. Alors, il existe b tel que $\pi - \tilde{\pi} = bP$ pour b quelconque. On obtient alors $c = \tilde{\alpha} - \alpha = ab$ et donc :

$$B = (c^{-1} \bmod q)(\pi - \tilde{\pi}) = (ab)^{-1}bP = a^{-1}P.$$

□

8.2.7 Traçage de traîtres en boîte noire

Pour des raisons pratiques, dans le traçage de traîtres, il est important que le décodeur pirate n'ait pas besoin d'être ouvert. Nous prouvons en effet que, pour notre schéma, nous pouvons faire le traçage de traîtres en boîte noire contre une coalition de deux usagers sous l'hypothèse $\mathsf{DBDH}^1-\mathsf{M}$. Nous construisons, de plus, un algorithme de traçage. Le résultat de sécurité est obtenu dans le modèle standard, où la fonction H est aléatoirement choisie dans une famille de fonctions de hachage universelles.

Théorème 57 *Étant donné une clef de chiffrement* pk *et une sous-clef de déchiffrement équivalente* $(\alpha, \pi = \beta P) \in \Pi_{\mathsf{pk}}$*, supposons qu'il existe un attaquant* $\mathcal{A}$ *qui peut produire un simulateur de déchiffrement* $\mathcal{S}$ *tel que : quand on lui donne un texte chiffré valable, il retourne correctement le texte clair correspondant ; quand on lui donne un texte chiffré « randomisé » de la forme* (kP, ak'^2P, d)*, où* $k, k' \overset{R}{\leftarrow} \mathbb{Z}_q$ *et* $d \overset{R}{\leftarrow} \mathcal{M}$*, il retourne une valeur différente de* $d \oplus H(g^{\alpha k^2 + a\beta k'^2})$ *avec probabilité* ε*. Alors, on peut résoudre le problème* $\mathsf{DBDH}^1-\mathsf{M}$ *avec probabilité* $\frac{1}{2} + \varepsilon/4$.

Preuve. On va construire un algorithme $\mathcal{B}$ qui, en utilisant $\mathcal{A}$ comme une sous-programme, résout le problème $\mathsf{DBDH}^1-\mathsf{M}$.

Une instance aléatoire du problème $\mathsf{DBDH}^1-\mathsf{M}$ $(P, A = aP, B = kP, X)$ (où $a, k \overset{R}{\leftarrow} \mathbb{Z}_q^\star$), où X est soit la solution du problème $\mathsf{DBDH}^1-\mathsf{M}$, soit aléatoirement choisi dans $\mathcal{G}_1$, est donnée à $\mathcal{B}$. $\mathcal{B}$ distingue ces deux cas de la façon suivante :

Implémentation : $\mathcal{B}$ choisit aléatoirement $\alpha \overset{R}{\leftarrow} \mathbb{Z}_q^\star$ et $\pi \overset{R}{\leftarrow} \mathcal{G}_1$, pose $Q = A$ et calcule $Z = g^\alpha \cdot e(Q, \pi)$. Puis, $\mathcal{B}$ définit la clef publique $\mathsf{pk} = (g, Q, Z)$ et transmet pk ainsi que la sous-clef de déchiffrement équivalente (α, π) à $\mathcal{A}$.

Challenge : $\mathcal{B}$ choisit aléatoirement $d \in \mathcal{M}$, construit un texte chiffré $(c_1 = B, c_2 = X, d)$ et le transmet à $\mathcal{A}$. Grâce aux choix aléatoires de (B, X, d) et de la fonction de hachage H dans la famille de fonctions de hachage universelles, le challenge (c_1, c_2, d) est un texte chiffré aléatoire.

Décision : Si l'attaquant $\mathcal{A}$ retourne $d \oplus H(e(\alpha c_1, c_1) \cdot e(\pi, X))$, $\mathcal{B}$ retourne aléatoirement 1 ou 0. Sinon $\mathcal{B}$ retourne 0 (X n'est probablement pas égal à ak^2P).

On considère le cas où $X = ak^2P$, le texte chiffré (kP, X, d) est un texte chiffré valide. Sous l'hypothèse du théorème sur l'attaquant $\mathcal{A}$, il retourne correctement le texte clair

$m = d \oplus H(e(\alpha c_1, c_1) \cdot e(\pi, X))$. Dans ce cas, l'algorithme $\mathcal{B}$ retourne aléatoirement 1 ou 0 et la probabilité que $\mathcal{B}$ donne une décision correcte est donc $1/2$.

Dans le cas où $X \neq ak^2P$, le texte chiffré (kP, X, d) est aléatoire et invalide. Par l'hypothèse du théorème sur l'attaquant $\mathcal{A}$, il retourne une valeur différente du texte clair espéré $d \oplus H(g^{\alpha k^2 + a\beta k'^2})$ avec probabilité ε. Si $\mathcal{A}$ retourne un texte clair différent de ce texte clair espéré, $\mathcal{B}$ sait répondre correctement que X n'est pas égal à ak^2P. Si $\mathcal{A}$ retourne le texte clair espéré (la probabilité de cet événement est au plus $1 - \varepsilon$), $\mathcal{B}$ répond aléatoirement 1 ou 0. Alors, la probabilité que $\mathcal{B}$ retourne une décision correcte est $\varepsilon + (1 - \varepsilon) \cdot 1/2 = 1/2 + \varepsilon/2$.

En combinant ces deux cas, on voit facilement que $\mathcal{B}$ peut retourner une décision correcte au problème $\mathsf{DBDH}^1{-}\mathsf{M}$ avec probabilité $1/2 + \varepsilon/4$. □

Intuitivement, ce théorème montre qu'un texte chiffré « randomisé », et donc invalide, et un texte chiffré valide sont indistinguables. Par conséquent, étant donné l'accès en boîte noire à un simulateur de déchiffrement $\mathcal{S}$ produit par un des deux usagers, on peut retrouver son fabriquant : on choisit aléatoirement $k, k' \overset{R}{\leftarrow} \mathbb{Z}_q^\star$ (on suppose que $k \neq k'$), et on pose $u_0 = \alpha_0 k^2 + a\beta_0 k'^2$ et $u_1 = \alpha_1 k^2 + a\beta_1 k'^2$. Avec une forte probabilité (plus grande que $1 - 2/q$), u_0 est différent de u_1. On soumet le texte chiffré randomisé (kP, ak'^2P, d) à $\mathcal{S}$. Si la sortie de $\mathcal{S}$ est d/g^{u_0} alors on peut déclarer que u_0 est le traître. Si la sortie est d/g^{u_1} alors u_1 est le traître. Sinon, *i.e.* la sortie est différente de ces deux valeurs, on conclut que les deux usagers se sont coalisés, d'où le corollaire suivant.

Corollaire 58 *Dans le schéma décrit ci-dessus, il est possible de faire le traçage de traîtres en boîte noire contre des attaquants actifs.*

8.2.8 Traçabilité publique

Maintenant, on considère une propriété additionnelle relativement intéressante : la traçabilité publique. Dans la section précédente, pour réaliser la procédure de traçage de traîtres en boîte noire, les deux clefs d'usagers (α_0, β_0) et (α_1, β_1) sont utilisées. Cependant, les sous-clefs de déchiffrement équivalentes suffisent pour le faire, voire moins : $(\alpha_0 P, \beta_0 P)$ et $(\alpha_1 P, \beta_1 P)$ suffisent. En effet, à partir de $k, k' \overset{R}{\leftarrow} \mathbb{Z}_q^\star$, on n'a pas vraiment besoin de u_0, u_1, mais seulement de g^{u_0} et g^{u_1} pour faire le traçage. Les valeurs de g^{u_0} et g^{u_1} peuvent être calculées de la manière suivante :

$$\begin{aligned} g^{u_0} &= e(\alpha_0 P, k^2 P) \cdot e(Q, k'^2(\beta_0 P)) \ ; \\ g^{u_1} &= e(\alpha_1 P, k^2 P) \cdot e(Q, k'^2(\beta_1 P)). \end{aligned}$$

Alors, si on rend publique les *informations de traçage* $(\alpha_0 P, \beta_0 P)$ et $(\alpha_1 P, \beta_1 P)$, n'importe qui peut faire le traçage et on peut donc alléger le travail du centre. Plus précisément, le traçage peut être délégué à une tierce personne et cette délégation n'exige aucune confiance en cette tierce partie, car les informations données n'aident ni à apprendre

des informations sur la clef secrète (a, z), ni à construire un décodeur pirate (sous une hypothèse additionnelle calculatoire).

Nous montrons ces propriétés de sécurité avec les théorèmes suivants.

Théorème 59 *Supposons que l'information de traçage soit rendue publique le schéma de chiffrement est sémantiquement sûr : dans le modèle de l'oracle aléatoire, la sécurité est fondée sur l'hypothèse* $\mathsf{CBDH}^2-\mathsf{E}$*, alors que dans le modèle standard, elle est fondée sur l'hypothèse* $\mathsf{DBDH}^2-\mathsf{E}$*.*

Preuve. On focalise sur le cas où la fonction H est aléatoirement choisie dans une famille de fonctions de hachage universelles. La preuve pour le cas où H est considérée comme un oracle aléatoire utilise la même technique que la preuve du théorème 54. Considérons un attaquant $\mathcal{A}$ contre le schéma de chiffrement selon une attaque IND-CPA. Supposons que $\mathcal{A}$ dispose d'un avantage ε, alors on peut résoudre le problème $\mathsf{DBDH}^2-\mathsf{E}$ avec probabilité $\varepsilon/2$. En effet, on construit un algorithme $\mathcal{B}$ qui, en utilisant $\mathcal{A}$ comme sous-programme, résout le problème $\mathsf{DBDH}^2-\mathsf{E}$.

Une instance aléatoire du problème $\mathsf{DBDH}^2-\mathsf{E}$ $(P, A = aP, B = kP, C = zP, D = ak^2P, U)$ (où $a, k \overset{R}{\leftarrow} \mathbb{Z}_q^\star$), où U est soit la solution du problème $\mathsf{CBDH}^2-\mathsf{E}$ soit aléatoirement choisi dans $\mathcal{G}_2$, est donnée à $\mathcal{B}$. $\mathcal{B}$ distingue ces deux cas de la façon suivante :

Implémentation : $\mathcal{B}$ choisit aléatoirement β_0, β_1 et calcule :

$$\alpha_0 P = zP - \beta_0 Q \qquad \alpha_1 P = zP - \beta_1 Q.$$

$\mathcal{B}$ définit la clef publique $\mathsf{pk} = (g, Q = A, Z = g^z = e(C, P))$. Il transmet pk et l'information de traçage $(\alpha_0 P, \beta_0 P, \alpha_1 P, \beta_1 P)$ à $\mathcal{A}$.

Challenger : Quand $\mathcal{A}$ retourne deux candidats m_0, m_1, $\mathcal{B}$ prend un bit aléatoire $b \in \{0, 1\}$ et transmet $(A, D, d = m_b \oplus H(U))$ comme challenge à $\mathcal{A}$.

Choix : Quand l'attaquant $\mathcal{A}$ devine $b' \in \{0, 1\}$, l'algorithme $\mathcal{B}$ retourne 1 (*i.e.* U est la solution au problème $\mathsf{DBDH}^2-\mathsf{E}$) si $b = b'$ et 0 sinon.

On observe que si U est la solution au problème $\mathsf{CBDH}^2-\mathsf{E}$, alors le texte chiffré C est un chiffrement de m_b. Sinon, comme H est aléatoirement choisie dans une famille de fonctions de hachage universelles, C est un chiffrement d'un message aléatoire, b est donc égal à b' avec probabilité $1/2$. Par un argument classique, on trouve que l'attaquant $\mathcal{B}$ dispose d'un avantage $\varepsilon/2$ pour résoudre le problème $\mathsf{CBDH}^2-\mathsf{E}$. □

Théorème 60 *Étant donné une clef de chiffrement* pk*, une sous-clef de déchiffrement équivalente* $(\alpha, \pi = \beta P) \in \Pi_{\mathsf{pk}}$ *et l'information publique de traçage* $(\alpha_0 P, \beta_0 P, \alpha_1 P, \beta_1 P)$*. Supposons qu'il existe un attaquant* $\mathcal{A}$ *qui peut fabriquer un simulateur de déchiffrement* $\mathcal{S}$ *tel que : quand on lui donne un texte chiffré valable, il retourne correctement le texte clair correspondant ; quand on lui donne un texte chiffré « randomisé » de la forme* (kP, ak'^2P, d)*, où* $k, k' \overset{R}{\leftarrow} \mathbb{Z}_q, d \overset{R}{\leftarrow} \mathcal{M}$*, il retourne une valeur différente de* $d \oplus H(g^{\alpha k^2 + a\beta k'^2})$ *avec probabilité* ε*. Alors, on peut résoudre le problème* $\mathsf{DBDH}^1-\mathsf{M}$ *avec probabilité* $\frac{1}{2} + \varepsilon/4$*.*

Preuve. On construit un algorithme $\mathcal{B}$ qui, en utilisant $\mathcal{A}$ comme sous-programme, résout le problème $\mathsf{DBDH}^1-\mathsf{M}$.

Une instance aléatoire du problème $\mathsf{DBDH}^1-\mathsf{M}$ $(P, A = aP, B = kP, X)$ (où $a, k \overset{R}{\leftarrow} \mathbb{Z}_q^\star$), où X est soit la solution du problème $\mathsf{DBDH}^1-\mathsf{M}$, soit aléatoirement choisi dans $\mathcal{G}_1$, est donnée à $\mathcal{B}$. $\mathcal{B}$ distingue ces deux cas de la façon suivante :

Implémentation : $\mathcal{B}$ choisit aléatoirement $\alpha_0, \beta_0, \beta_1 \overset{R}{\leftarrow} \mathbb{Z}_q^\star$ et calcule :

$$zP = \alpha_0 P + \beta_0 A \qquad \alpha_1 P = zP - \beta_1 A \qquad \pi_0 = \beta_0 P \qquad Z = e(P, zP).$$

$\mathcal{B}$ pose $Q = A$, $\mathsf{pk} = (g, Q, Z)$ et une sous-clef de déchiffrement équivalente (α_0, π_0), et l'information de traçage $(\alpha_0 P, \beta_0 P, \alpha_1 P, \beta_1 P)$. La sous-clef de déchiffrement équivalente (α_0, π_0) peut être considérée comme un élément aléatoire dans l'ensemble Π_{pk}. $\mathcal{B}$ transmet pk, la sous-clef de déchiffrement équivalente (α_0, π_0) et l'information de traçage $(\alpha_0 P, \beta_0 P, \alpha_1 P, \beta_1 P)$ à $\mathcal{A}$.

Challenge : $\mathcal{B}$ choisit aléatoirement $d \in \mathcal{M}$, construit un texte chiffré $(c_1 = B, c_2 = X, d)$ et le transmet à $\mathcal{A}$. Grâce aux choix aléatoires de (B, X, d) et de la fonction de hachage H dans la famille des fonctions de hachage universelles, le challenge (c_1, c_2, d) est un texte chiffré aléatoire.

Décision : Si l'attaquant $\mathcal{A}$ retourne $d \oplus H(e(\alpha_0 c_1, c_1) \cdot e(\pi_0, X))$, $\mathcal{B}$ retourne aléatoirement 1 ou 0. Sinon, $\mathcal{B}$ retourne 0 (X n'est probablement pas égale à ak^2P).

On considère le cas où $X = ak^2P$, le texte chiffré (kP, X, d) est valide. Sous l'hypothèse du théorème sur l'attaquant $\mathcal{A}$, il retourne correctement le texte clair $m = d \oplus H(e(\alpha_0 c_1, c_1) \cdot e(\pi_0, X))$. Dans ce cas, l'algorithme $\mathcal{B}$ retourne aléatoirement 1 ou 0 et la probabilité que $\mathcal{B}$ donne une décision correcte est donc égale à $1/2$.

Dans le cas où $X \neq ak^2P$, le texte chiffré (kP, X, d) est aléatoire et invalide. Sous l'hypothèse du théorème sur l'attaquant $\mathcal{A}$, il retourne une valeur différente du texte clair espéré $d \oplus H(g^{\alpha k^2 + a\beta k'^2})$ avec probabilité ε. Si $\mathcal{A}$ retourne un texte clair différent de ce texte clair espéré, $\mathcal{B}$ sait répondre correctement que X n'est pas égal à ak^2P. Si $\mathcal{A}$ retourne le texte clair espéré (la probabilité de cet événement est au plus $1 - \varepsilon$), $\mathcal{B}$ répond aléatoirement 1 ou 0. Alors, la probabilité que $\mathcal{B}$ retourne une décision correcte est $\varepsilon + (1 - \varepsilon) \cdot 1/2 = 1/2 + \varepsilon/2$.

En combinant ces deux cas, on voit facilement que $\mathcal{B}$ peut retourner une décision correcte au problème $\mathsf{DBDH}^1-\mathsf{M}$ avec probabilité $1/2 + \varepsilon/4$. □

8.3 Schéma à plusieurs usagers

8.3.1 Description

Considérons $C = \{\omega_1, \ldots, \omega_N\}$, un code $(N, c, \ell, \varepsilon)$ résistant aux coalitions. Ce code contient N mots de code sur l bits et sur l'alphabet $\{0, 1\}$. Il résiste aux coalitions de c

usagers, *i.e.* permet de construire un algorithme de traçage pour trouver un des traîtres avec probabilité de succès $1-\varepsilon$, à partir d'un mot de code « faisable » fabriqué par une coalition de c usagers. Les détails des codes résistants aux coalitions sont décrits dans [25]. Le cas multi-usagers est simplement une *ℓ-instantiation* de schémas de deux usagers. En effet, nous construisons un système multi-usagers en utilisant C-un code $(N, c, \ell, \varepsilon)$ résistant aux coalitions pour bien combiner ℓ systèmes à deux usagers S_1, S_2, ..., S_ℓ :

Implémentation : Étant donné les paramètres de sécurité k, c et ε :

Étape 1 : Générer un nombre premier q de k bits et deux groupes $\mathcal{G}_1$, $\mathcal{G}_2$ d'ordre q ainsi qu'une application bilinéaire admissible $e : \mathcal{G}_1 \times \mathcal{G}_1 \to \mathcal{G}_2$. Choisir aléatoirement un générateur $P \overset{R}{\leftarrow} \mathcal{G}_1$ et poser $g = e(P, P)$, qui est un générateur du groupe $\mathcal{G}_2$.

Étape 2 : Générer $C = \{\omega_1, .., \omega_N\}$, un code $(N, c, \ell, \varepsilon)$ résistant aux coalitions.

Étape 3 : Prendre aléatoirement des éléments $a, z_j \overset{R}{\leftarrow} \mathbb{Z}_q^\star$, et poser $Q = aP$, $Z_j = g^{z_j}$, pour $j = 1, \ldots, \ell$.

Étape 4 : Choisir une fonction $H : \mathcal{G}_1 \to \mathcal{M}$.

Les paramètres du système sont $\mathsf{params} = (q, \mathcal{G}_1, \mathcal{G}_2, e, P, H)$. Remarquons que ces paramètres sont communs à tous les systèmes à deux usagers S_1, S_2, ..., S_ℓ.

Clef secrète du centre : l'élément a, et l'ensemble $(z_j)_{j=1,\ldots,\ell}$

Clef de chiffrement : la combinaison des clefs de chiffrement des ℓ schémas à deux usagers : $\mathsf{pk} = (g, Q, \{Z_j = g^{z_j}\}_{j=1,\ldots,\ell})$

Clef d'usager : l'usager u_i (pour $i \in \mathbb{Z}_N$) est associé à un mot de code $\omega_i \in C$ et une « représentation » correspondante $(\alpha_{\omega_{i,j},j}, \beta_{\omega_{i,j},j})$ de g^{z_j} dans la base (g, g^a), où $\omega_{i,j}$ est le j-ième bit du mot de code ω_i. Rappelons que $(\alpha_{b,j}, \beta_{b,j})$ est une « représentation » de g^{z_j} dans la base (g, g^a). Remarquons aussi que l'autorité génère ces clefs pour les usagers mais ne les leur donne pas. Chaque usager reçoit une sous-clef de déchiffrement équivalente, décrite ci-dessous.

Sous-clef de déchiffrement équivalente : l'usager u_i (pour $i \in \mathbb{Z}_N$) reçoit une sous-clef de déchiffrement équivalente $\Pi_i = (\Pi_{\omega_{i,1},1}, \ldots, \Pi_{\omega_{i,l},\ell})$. Plus précisément, pour $j = 1, \ldots, \ell$,

$$\Pi_{\omega_{i,j},j} = (\alpha_{\omega_{i,j},j}, \pi_{\omega_{i,j},j} = \beta_{\omega_{i,j},j} P).$$

Algorithme de chiffrement : L'espace des textes clairs du système est $\mathcal{M}^\ell$. À partir de l'entrée $(m_1, \ldots, m_\ell)$, l'algorithme de chiffrement génère un aléa $k \in \mathbb{Z}_q$ et retourne un texte chiffré $(c_1, c_2, d_1, \ldots, d_\ell)$ dans $\mathcal{G}_1 \times \mathcal{G}_1 \times \mathcal{G}_2^\ell$, où : $c_1 = kP$, $c_2 = k^2 aP$ et $d_j = m_j \oplus H(Z_j^{k^2})$.

Algorithme de déchiffrement : Sur un texte chiffré $(c_1, c_2, d_1, \ldots, d_\ell)$, l'usager u_i calcule, grâce à sa sous-clef de déchiffrement équivalente, $Z_j^{k^2} = e(\alpha_{\omega_{i,j},j} c_1, c_1) \cdot e(\pi_{\omega_{i,j},j}, c_2)$ et $m_j = d_j \oplus H(Z_j^{k^2})$.

Pour l'analyse de sécurité, on peut utiliser l'hypothèse suivante, de [73] : l'*hypothèse à seuil* selon la quelle « un décodeur pirate qui retourne correctement une fraction p du texte clair de longueur λ, où $(1-p)$ est une fonction non-négligeable en λ, est inutile ». Cependant,

comme mentionné dans [73], en utilisant une transformation « *all-or-nothing* » [106, 28], cette hypothèse n'est plus nécessaire.

Proposition 61 *Etant donné ℓ systèmes de deux usagers. La coalition de deux usagers dans $(\ell - 1)$ systèmes n'affaiblit pas la sécurité du système restant.*

Preuve. Supposons qu'il existe un attaquant qui, disposant d'une information I du système S_1 et de toutes les informations des systèmes $S_2, \ldots, S_\ell$, peut avoir un avantage ε contre le système S_1 (pour un but G quelconque). Alors, on peut construire un algorithme $\mathcal{B}$ qui, disposant seulement d'une information I du système S_1, peut avoir un avantage ε contre le système S_1.

L'idée est que l'algorithme $\mathcal{B}$ peut simuler toutes les informations concernant cette coalition : étant donné les paramètres $\mathsf{params} = (q, \mathcal{G}_1, \mathcal{G}_2, e, P, H)$ et l'information publique (g, Q, Z_1), $\mathcal{B}$ peut facilement générer des informations pour les systèmes S_j, $j = 2, 3, \ldots, \ell$:
- il génère aléatoirement des éléments $\alpha_{0,j}$, $\beta_{0,j}$, $\alpha_{1,j}$;
- il calcule $Z_j = e(\alpha_{0,j}P, P) \cdot e(\beta_{0,j}P, Q)$;
- il crée les sous-clefs de déchiffrement équivalentes par $\pi_{0,j} = \beta_{0,j}P$ et $\pi_{1,j} = \alpha_{0,j}P + \beta_{0,j}Q - \alpha_{1,j}P$.

□

Cette proposition, combinée avec le fait que C est un code $(N, c, \ell, \varepsilon)$ résistant aux coalitions, conduit au corollaire suivant :

Corollaire 62 *Le schéma décrit ci-dessus est un schéma à N usagers qui supporte le traçage de traîtres.*

En ce qui concerne la traçabilité publique, avec les informations publiques, n'importe qui peut retrouver le mot de code associé au décodeur pirate. Cette phase est interactive et est donc la plus coûteuse. Cependant, l'utilisation des codes classiques résistant aux coalitions ne permet pas de faire publiquement tout le traçage car la phase de traçage effectif des traîtres à partir d'un mot de code pirate est secrète. Cette phase est une procédure *off-line* et doit être réalisée par une autorité de confiance.

8.3.2 Comparaison de l'efficacité avec le schéma de Kiayias-Yung

Dans le schéma KY, le taux de texte chiffré est 3. Alors que dans le nôtre, il est asymptotiquement égale à 1. On peut se demander pourquoi la construction ci-dessus n'est pas applicable au schéma de Kiayias et Yung ?

La raison est que dans notre schéma à deux usagers, même une coalition de deux usagers ne révèle pas d'information sur a, alors que dans le schéma de de Kiayias et Yung, une telle coalition révèle complètement a. Par conséquent, dans le cas multi-usagers de

leur schéma, si on utilise le même a pour les ℓ schémas à deux usagers, la coalition de deux usagers peut révéler cette valeur de a, puis toutes les z_i. Ceci permet évidemment une construction anonyme d'un décodeur pirate. C'est pourquoi ils ont dû utiliser une valeur différente de a dans chaque sous-schéma à deux usagers.

8.4 Conclusion

Nous avons proposé un schéma qui améliore le schéma de Kiayias et Yung selon plusieurs critères : premièrement, les taux de transmission sont réduits, « approchées » des valeurs optimales ; deuxièmement nous introduisons une fonctionnalité intéressante : la *traçabilité publique*. La possibilité d'obtenir la propriété complète de traçabilité publique dans le cas multi-usagers reste cependant un problème ouvert.

Conclusion

Dans cette thèse, nous avons étudié trois aspects importants de la sécurité prouvée : les notions de sécurité et leurs relations entre elles ; la construction de schémas pratiques et formellement prouvés sûrs ; de nouvelles fonctionnalités pour la cryptographie. Nous profitons de cette conclusion pour résumer les résultats les plus importants et aussi pour proposer un certain nombre de problèmes ouverts pour les travaux à venir.

Contribution Dans la première partie, nous étudions les relations entre les notions de sécurité, à la fois pour le chiffrement asymétrique et pour le chiffrement symétrique. Dans le cadre du chiffrement asymétrique, nous avons montré qu'une requête non-adaptative (par rapport au challenge) de déchiffrement, de façon contre-intuitive, ne peut être remplacée par une, ni même plusieurs requêtes adaptatives de déchiffrement. Nous montrons ensuite que la situation est totalement différente dans le cadre du chiffrement symétrique et déterministe : un attaquant adaptatif n'a pas une puissance supérieure par rapport à un attaquant non-adaptatif, sous l'hypothèse que le chiffrement et le déchiffrement résistent aux attaques non-adaptatives. Nous mettons aussi en évidence les relations entre la notion d'indistinguabilité et les notions conventionnelles utilisées dans les chiffrements par bloc, *i.e.* famille de permutations (super) pseudo-aléatoires.

Dans une deuxième partie, nous proposons de nouveaux schémas efficaces et prouvés sûrs pour le chiffrement asymétrique dans le modèle de la permutation aléatoire (construction FDP) et de l'oracle aléatoire (construction OAEP 3 tours). Nous présentons une nouvelle classe de schémas de chiffrement $\mathsf{IND} - \mathsf{CCA2}$: des schémas sans redondance, où tout chiffré correspond au chiffrement d'un texte clair. Notons que jusqu'à présent, les redondances étaient nécessaires pour prouver la sécurité d'un schéma contre des attaques à chiffrés choisis adaptatives. La suppression des redondances améliore non seulement l'efficacité des schémas (en réduisant l'expansion) mais aide également à éviter d'éventuelles attaques en pratique comme des attaques « side-channel » car tous les chiffrés sont déchiffrés selon le même mécanisme. Nous élargissons nos constructions pour construire de nouveaux paddings universels qui utilisent une paire de (clés publique/ clés secrète) simultanément pour le chiffrement et pour la signature. De plus, en exploitant un seul champ réservé, soit à l'aléa (lors d'un chiffrement), soit à la redondance (lors d'une signature), l'efficacité s'en trouve améliorée.

Dans la dernière partie, nous abordons le problème de la diffusion de données chiffrées

dont un des plus grands soucis est de retrouver des traîtres qui coopèrent pour produire des décodeurs pirates. Nous avons introduit une nouvelle fonctionnalité pour ce problème : la traçabilité publique, *i.e.* n'importe qui peut réaliser la phase de traçage de traîtres à partir des informations publiques uniquement. Notre schéma de base atteint cette propriété. Dans le schéma général, cette propriété est partiellement atteinte dans la mesure où la phase interactive avec le décodeur pirate peut être publiquement achevée mais la phase non interactive (ou « off-line ») nécessite l'intervention du centre. Même dans ce cas, cette propriété « partielle » permet au centre de déléguer efficacement la tâche la plus coûteuse du traçage. D'un point de vue de l'efficacité, notre schéma général a atteint un taux entre la taille des données chiffrées et celle des données originelles qui est asymptotiquement optimal.

Problèmes ouverts Les résultats dans cette thèse pourront évidemment être améliorés. De nombreux problèmes restent ouverts.

- Concernant le chiffrement par bloc, nous proposons d'étudier non seulement la sécurité du schéma, mais également celle de son inverse (où les algorithmes de chiffrement et de déchiffrement sont échangés). A titre d'exemple, une preuve de la résistance du déchiffrement face aux attaques non-adaptatives permet de mettre en équivalence les attaques adaptatives et les attaques non-adaptatives, l'indistinguabilité contre les attaques non-adaptatives et la propriété super-pseudo-aléatoire du chiffrement.
- Pour le chiffrement asymétrique, un problème intéressant est de construire un schéma $\mathsf{IND}-\mathsf{CCAO2}$ simple (*i.e.* plus simple que les schémas $\mathsf{IND}-\mathsf{CCA2}$). Un tel schéma serait également $\mathsf{NM}-\mathsf{CPA}$. Notons que l'on ne connaît pas encore de construction simple de schémas non-malléables même contre des attaques à clairs choisis.
- Pour le chiffrement sans redondance, la construction d'un schéma $\mathsf{IND}-\mathsf{CCA2}$ dans le modèle standard reste un problème ouvert. S'il n'existait pas de telle construction, il s'agirait alors d'une primitive non-artificielle instantiable dans le modèle de l'oracle aléatoire, et ininstantiable dans le modèle standard.
- Pour la diffusion de données chiffrées, nous proposons de construire un schéma général qui atteindrait pleinement la traçabilité publique. Les codes résistants aux coalitions semblent difficiles à conjuguer avec la traçabilité publique, mais peut-être cette propriété peut-elle être atteinte avec d'autres codes.

Bibliographie

[1] M. Agrawal, N. Kayal, and N. Saxena. PRIMES is in P. *Annals of Mathematics*, 160:781–793, 2004.

[2] J. H. An, Y. Dodis, and T. Rabin. On the Security of Joint Signature and Encryption. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 83–107, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer-Verlag, Berlin, Germany.

[3] P. S. L. M. Barreto. The Pairing-Based Crypto Lounge. Available from `http://paginas.terra.com.br/informatica/paulobarreto/pblounge.html`.

[4] M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-Privacy in Public-Key Encryption. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582, Gold Coast, Australia, December 9–13, 2001. Springer-Verlag, Berlin, Germany.

[5] M. Bellare, A. Boldyreva, and A. Palacio. An Uninstantiable Random-Oracle-Model Scheme for a Hybrid-Encryption Problem. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 171–188, Interlaken, Switzerland, May 2–6, 2004. Springer-Verlag, Berlin, Germany.

[6] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A Concrete Security Treatment of Symmetric Encryption. In *38th Annual Symposium on Foundations of Computer Science*, pages 394–403, Miami Beach, Florida, October 19–22, 1997. IEEE Computer Society Press.

[7] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations Among Notions of Security for Public-Key Encryption Schemes. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 26–45, Santa Barbara, CA, USA, August 23–27, 1998. Springer-Verlag, Berlin, Germany.

[8] M. Bellare, S. Halevi, A. Sahai, and S. P. Vadhan. Many-to-One Trapdoor Functions and Their Relation to Public-Key Cryptosystems. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 283–298, Santa Barbara, CA, USA, August 23–27, 1998. Springer-Verlag, Berlin, Germany.

[9] M. Bellare and P. Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *ACM CCS 93: 1st Conference on Computer and Com-*

munications Security, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press.

[10] M. Bellare and P. Rogaway. Optimal Asymmetric Encryption – How to Encrypt with RSA. In Alfredo De Santis, editor, *Advances in Cryptology – EUROCRYPT'94*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111, Perugia, Italy, May 9–12, 1994. Springer-Verlag, Berlin, Germany.

[11] M. Bellare and P. Rogaway. The Exact Security of Digital Signatures: How to Sign with RSA and Rabin. In Ueli M. Maurer, editor, *Advances in Cryptology – EUROCRYPT'96*, volume 1070 of *Lecture Notes in Computer Science*, Saragossa, Spain, May 12–16, 1996. Springer-Verlag, Berlin, Germany.

[12] M. Bellare and A. Sahai. Non-malleable Encryption: Equivalence between Two Notions, and an Indistinguishability-Based Characterization. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 519–536, Santa Barbara, CA, USA, August 15–19, 1999. Springer-Verlag, Berlin, Germany.

[13] D. Bleichenbacher. Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS #1. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 1–12, Santa Barbara, CA, USA, August 23–27, 1998. Springer-Verlag, Berlin, Germany.

[14] L. Blum, M. Blum, and M. Shub. A Simple Unpredictable Pseudo-Random Number Generator. *SIAM Journal on Computing*, 15(2):364–383, May 1986.

[15] M. Blum, P. Feldman, and S. Micali. Non-Interactive Zero-Knowledge and its Applications. In *20th Annual ACM Symposium on Theory of Computing*, pages 103–112, Chicago, Illinois, USA, May 2–4, 1988. ACM Press.

[16] M. Blum and S. Goldwasser. An Efficient Probabilistic Public-Key Encryption Scheme Which Hides All Partial Information. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 289–302, Santa Barbara, CA, USA, August 19–23, 1985. Springer-Verlag, Berlin, Germany.

[17] M. Blum and S. Micali. How to Generate Cryptographically Strong Sequences of Pseudorandom Bits. In *23rd Annual Symposium on Foundations of Computer Science*, pages 112–117, Chicago, Illinois, November 3–5, 1982. IEEE Computer Society Press.

[18] M. Blum and S. Micali. How to Generate Cryptographically Strong Sequences of Pseudorandom Bits. *SIAM Journal on Computing*, 13(4):850–864, 1984.

[19] D. Boneh. Simplified OAEP for the RSA and Rabin Functions. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 275–291, Santa Barbara, CA, USA, August 19–23, 2001. Springer-Verlag, Berlin, Germany.

[20] D. Boneh and X. Boyen. Efficient Selective-ID Secure Identity Based Encryption Without Random Oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Com-*

puter Science, pages 223–238, Interlaken, Switzerland, May 2–6, 2004. Springer-Verlag, Berlin, Germany.

[21] D. Boneh and X. Boyen. Secure Identity Based Encryption Without Random Oracles. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 443–459, Santa Barbara, CA, USA, August 15–19, 2004. Springer-Verlag, Berlin, Germany.

[22] D. Boneh and M. K. Franklin. An Efficient Public Key Traitor Tracing Scheme. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 338–353, Santa Barbara, CA, USA, August 15–19, 1999. Springer-Verlag, Berlin, Germany.

[23] D. Boneh and M. K. Franklin. Identity-Based Encryption from the Weil Pairing. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229, Santa Barbara, CA, USA, August 19–23, 2001. Springer-Verlag, Berlin, Germany.

[24] D. Boneh, H. Shacham, and B. Lynn. Short Signatures from the Weil Pairing. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532, Gold Coast, Australia, December 9–13, 2001. Springer-Verlag, Berlin, Germany.

[25] D. Boneh and J. Shaw. Collusion Secure Fingerprinting for Digital Data. *IEEE Transactions on Information Theory*, 44(5):1897–1905, 1998.

[26] Dan Boneh and Xavier Boyen. Short Signatures Without Random Oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 56–73, Interlaken, Switzerland, May 2–6, 2004. Springer-Verlag, Berlin, Germany.

[27] D. R. L. Brown. The Exact Security of ECDSA. Contributions to IEEE P1363a, January 2001. `http://grouper.ieee.org/groups/1363/`.

[28] R. Canetti, Y. Dodis, S. Halevi, E. Kushilevitz, and A. Sahai. Exposure-Resilient Functions and All-or-Nothing Transforms. In Bart Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 453–469, Bruges, Belgium, May 14–18, 2000. Springer-Verlag, Berlin, Germany.

[29] R. Canetti, O. Goldreich, and S. Halevi. The Random Oracles Methodology, Revisited. In *30th Annual ACM Symposium on Theory of Computing*, pages 209–218, Dallas, Texas, USA, May 23–26, 1998. ACM Press.

[30] R. Canetti, O. Goldreich, and S. Halevi. On the Random-Oracle Methodology as Applied to Length-Restricted Signature Schemes. In *Proc. of the 1st Theory of Cryptography Conference - TCC '04*, LNCS 2951, pages 40–57, Berlin, 2004. Springer-Verlag.

[31] R. Canetti, H. Krawczyk, and J. B. Nielsen. Relaxing Chosen-Ciphertext Security. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 565–582, Santa Barbara, CA, USA, August 17–21, 2003. Springer-Verlag, Berlin, Germany.

[32] H. Chabanne, D. H. Phan, and D. Pointcheval. Public Traceability in Traitor Tracing Schemes. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 542–558, Aarhus, Denmark, May 22–26, 2005. Springer-Verlag, Berlin, Germany. Full version available from `http://www.di.ens.fr/users/phan/`.

[33] B. Chevallier-Mames, D. H. Phan, and D. Pointcheval. Optimal Asymmetric Encryption and Signature Paddings. In *Proceedings of ACNS '05*, volume LNCS 3531, pages 254–268, Berlin, 2005. Springer-Verlag. Full version available from `http://www.di.ens.fr/users/phan/`.

[34] J.-S. Coron. On the Exact Security of Full Domain Hash. In Mihir Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 229–235, Santa Barbara, CA, USA, August 20–24, 2000. Springer-Verlag, Berlin, Germany.

[35] J.-S. Coron. Optimal Security Proofs for PSS and other Signature Schemes. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer-Verlag, Berlin, Germany.

[36] J.-S. Coron, M. Joye, D. Naccache, and P. Paillier. Universal Padding Schemes for RSA. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 226–241, Santa Barbara, CA, USA, August 18–22, 2002. Springer-Verlag, Berlin, Germany.

[37] R. Cramer and V. Shoup. A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO'98*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25, Santa Barbara, CA, USA, August 23–27, 1998. Springer-Verlag, Berlin, Germany.

[38] R. Cramer and V. Shoup. Signature Scheme based on the Strong RSA Assumption. In *ACM CCS 99: 6th Conference on Computer and Communications Security*, pages 46–51, Kent Ridge Digital Labs, Singapore, November 1–4, 1999. ACM Press.

[39] R. Cramer and V. Shoup. Universal Hash Proofs and a Paradigm for Chosen-Ciphertext Secure Public Key Encryption. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 45–64, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer-Verlag, Berlin, Germany.

[40] A. Desai and S. Miner. Concrete Security Characterization of PRFs and PRPs: Reduction and Applications. In Tatsuaki Okamoto, editor, *Advances in Cryptology – ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 503–516, Kyoto, Japan, December 3–7, 2000. Springer-Verlag, Berlin, Germany.

[41] W. Diffie and M. E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, IT–22(6):644–654, November 1976.

[42] Y. Dodis and L. Reyzin. On the Power of Claw-Free Permutation. In Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano, editors, *SCN 02: 3rd International*

Conference on Security in Communication Networks, volume 2576 of *Lecture Notes in Computer Science*, pages 55–73, Amalfi, Italy, September 12–13, 2002. Springer-Verlag, Berlin, Germany.

[43] D. Dolev, C. Dwork, and M. Naor. Non-Malleable Cryptography. In *23rd Annual ACM Symposium on Theory of Computing*, pages 542–552, New Orleans, Louisiana, USA, May 6–8, 1991. ACM Press.

[44] D. Dolev, C. Dwork, and M. Naor. Non-Malleable Cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.

[45] T. El Gamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18, Santa Barbara, CA, USA, August 19–23, 1985. Springer-Verlag, Berlin, Germany.

[46] U. Feige, A. Fiat, and A. Shamir. Zero-Knowledge Proofs of Identity. In *19th Annual ACM Symposium on Theory of Computing*, pages 210–217, New York City,, New York, USA, May 25–27, 1987. ACM Press.

[47] U. Feige, A. Fiat, and A. Shamir. Zero-Knowledge Proofs of Identity. *Journal of Cryptology*, 1:77–95, 1988.

[48] U. Feige, D. Lapidot, and A. Shamir. Multiple Non-Interactive Zero Knowledge Proofs based on a Single Random String. In *31st Annual Symposium on Foundations of Computer Science*, pages 308–317, St. Louis, Missouri, October 22–24, 1990. IEEE Computer Society Press.

[49] A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions of Identification and Signature Problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology – CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194, Santa Barbara, CA, USA, August 1987. Springer-Verlag, Berlin, Germany.

[50] G. Frey and H. G. Rück. A Remark Concerning m-Divisibility and the Discrete Logarithm in the Divisor Class Group of Curves. *Mathematics of Computation*, 62:865–874, 1994.

[51] E. Fujisaki and T. Okamoto. How to Enhance the Security of Public-Key Encryption at Minimum Cost. In *PKC '99*, LNCS 1560, pages 53–68. Springer-Verlag, Berlin, 1999.

[52] E. Fujisaki and T. Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 537–554, Santa Barbara, CA, USA, August 15–19, 1999. Springer-Verlag, Berlin, Germany.

[53] E. Fujisaki and T. Okamoto. How to Enhance the Security of Public-Key Encryption at Minimum Cost. *IEICE Transaction of Fundamentals of Electronic Communications and Computer Science*, E83-A(1):24–32, January 2000.

[54] E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. RSA-OAEP is Secure under the RSA Assumption. In Michael J. Wiener, editor, *Advances in Cryptology –*

CRYPTO'99, volume 1666 of *Lecture Notes in Computer Science*, pages 260–274, Santa Barbara, CA, USA, August 15–19, 1999. Springer-Verlag, Berlin, Germany.

[55] O. Goldreich, S. Goldwasser, and S. Micali. How to Construct Random Functions. In *25th Annual Symposium on Foundations of Computer Science*, pages 464–479, Singer Island, Florida, October 24–26, 1984. IEEE Computer Society Press.

[56] O. Goldreich, S. Goldwasser, and S. Micali. How to Construct Random Functions. *Journal of the ACM*, 33:792–807, 1986. Full version of [55].

[57] O. Goldreich and L.A. Levin. A Hard-Core Predicate for all One-Way Functions. In *Proc. of the 21st STOC*, pages 25–32. ACM Press, New York, 1989.

[58] O. Goldreich, S. Micali, and A. Wigderson. Proofs That Yield Nothing but Their Validity and a Methodology of Cryptographic Protocol Design. In *Proc. of the 27th FOCS*, pages 174–187. IEEE, New York, 1986.

[59] S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28:270–299, 1984.

[60] S. Goldwasser, S. Micali, and R. Rivest. A "Paradoxical" Solution to the Signature Problem. In *25th Annual Symposium on Foundations of Computer Science*, pages 441–448, Singer Island, Florida, October 24–26, 1984. IEEE Computer Society Press.

[61] S. Goldwasser, S. Micali, and R. Rivest. A Digital Signature Scheme Secure Against Adaptative Chosen-Message Attacks. *SIAM Journal of Computing*, 17(2):281–308, April 1988.

[62] S. Goldwasser and Y. Taumann. On the (In)Security of the Fiat-Shamir Paradigm. In *44th Annual Symposium on Foundations of Computer Science*, pages 102–115, Cambridge, Massachusetts, USA, October 11–14, 2003. IEEE Computer Society Press.

[63] J. Håstad. Pseudo-Random Generators under Uniform Assumptions. In *22nd Annual ACM Symposium on Theory of Computing*, Baltimore, Maryland, USA, May 14–16, 1990. ACM Press.

[64] J. Håstad, R. Impagliazzo, L. Levin, and M. Luby. A Pseudorandom Generator from any One-Way Function. *SIAM Journal of Computing*, 28(4):1364–1396, 1999.

[65] I. Impagliazzo, L. Levin, and M. Luby. Pseudo-Random Generation from One-Way Functions. In *Proc. of the 21st STOC*, pages 12–24. ACM Press, New York, 1989.

[66] R. Impagliazzo and S. Rudich. Limits on the Provable Consequences of One-Way Permutations. In *21st Annual ACM Symposium on Theory of Computing*, pages 44–61, Seattle, Washington, USA, May 15–17, 1989. ACM Press.

[67] A. Joux. A One-Round Protocol for Tripartite Diffie-Hellman. In *Algorithmic Number Theory Symposium (ANTS IV)*, LNCS 1838, pages 385–394. Springer-Verlag, Berlin, 2000.

[68] A. Joux. A One Round Protocol for Tripartite Diffie-Hellman. *Journal of Cryptology*, 17(4):263–276, 2004.

[69] A. Joux and K. Nguyen. Separating Decision Diffie-Hellman from Computational Diffie-Hellman in Cryptographic Groups. *Journal of Cryptology*, 16(4):239–247, 2003.

[70] J. Kahn, M. Saks, and C. Smyth. A Dual Version of Reimer's Inequality and a Proof of Rudich's Conjecture. In *Proceedings of the 15th Annual IEEE Conference on Computational Complexity*, pages 98–103, Washington, DC, USA, 2000. IEEE Computer Society.

[71] J. Katz and N. Wang. Efficiency Improvements for Signature Schemes with Tight Security Reductions. In *ACM CCS 03: 10th Conference on Computer and Communications Security*, pages 155–164, Washington D.C., USA, October 27–30, 2003. ACM Press.

[72] J. Katz and M. Yung. Complete Characterization of Security Notions for Probabilistic Private-Key Encryption. In *Proc. of the 32nd STOC*. ACM Press, New York, 2000.

[73] A. Kiayias and M. Yung. Traitor Tracing with Constant Transmission Rate . In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 450–465, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer-Verlag, Berlin, Germany.

[74] A. Kiayias and M. Yung. Breaking and Repairing Asymmetric Public-Key Traitor Tracing. In J. Feigenbaum, editor, *ACM Workshop in Digital Rights Management – DRM 2002*, volume LNCS 2696, pages 32–50. Springer, 2003.

[75] N. Koblitz. Elliptic Curve Cryptosystems. *Mathematics of Computation*, 48(177):203–209, January 1987.

[76] Y. Komano and K. Ohta. Efficient Universal Padding Schemes for Multiplicative Trapdoor One-Way Permutation. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 366–382, Santa Barbara, CA, USA, August 17–21, 2003. Springer-Verlag, Berlin, Germany.

[77] A. Lenstra and E. Verheul. Selecting Cryptographic Key Sizes. In Hideki Imai and Yuliang Zheng, editors, *PKC 2000: 3rd International Workshop on Theory and Practice in Public Key Cryptography*, volume 1751 of *Lecture Notes in Computer Science*, pages 446–465, Melbourne, Victoria, Australia, January 18–20, 2000. Springer-Verlag, Berlin, Germany.

[78] M. Luby and Ch. Rackoff. How to Construct Pseudorandom Permutations from Pseudorandom Functions. *SIAM Journal of Computing*, 17(2):373–386, 1988.

[79] J. Manger. A Chosen Ciphertext Attack on RSA Optimal Asymmetric Encryption Padding (OAEP) as Standardized in PKCS #1. In *Advances in Cryptology – CRYPTO 2001*, LNCS 2139, pages 230–238. Springer-Verlag, Berlin, 2001.

[80] A. J. Menezes, T. Okamoto, and S. A. Vanstone. Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field. *IEEE Transactions on Information Theory*, 39:1639–1646, 1993.

[81] R. Merkle and M. Hellman. Hiding Information and Signatures in Trapdoor Knapsacks. *SIAM Journal on Computing*, IT-24(5):525–530, 1978.

[82] S. Micali, C. Rackoff, and R. Sloan. The Notion of Security for Probabilistic Cryptosystems. *SIAM Journal on Computing*, 17(2):412–426, April 1988. Special issue on cryptography.

[83] V. Miller. Uses of Elliptic Curves in Cryptography. In *Advances in Cryptology – CRYPTO'85*, LNCS 218, pages 417–426. Springer-Verlag, Berlin, 1986.

[84] S. Mitsunari, R. Sakai, and M. Kasahara. A New Traitor Tracing Schemes. *IEICE Trans. Fundamentals*, E85-A(2), 2002.

[85] L. J. Mordell. On the Rational Solutions of the Indeterminate Equations of the Third and Forth Degree. In *Cambridge Philos. Soc.*, pages 179–192, 1922.

[86] D. Naccache and J. Stern. Signing on a Postcard. In *Financial Cryptography '00*, LNCS 1962. Springer-Verlag, Berlin, 2001.

[87] M. Naor and M. Yung. Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attack. In *22nd Annual ACM Symposium on Theory of Computing*, pages 427–437, Baltimore, Maryland, USA, May 14–16, 1990. ACM Press.

[88] J. B. Nielsen. Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-committing Encryption Case. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 111–126, Santa Barbara, CA, USA, August 18–22, 2002. Springer-Verlag, Berlin, Germany.

[89] NIST. Digital Signature Standard (DSS). Federal Information Processing Standards PUBlication 186, November 1994.

[90] NIST. Secure Hash Standard (SHS). Federal Information Processing Standards PUBlication 180–1, April 1995.

[91] NIST. Descriptions of SHA–256, SHA–384, and SHA–512. Available from `http://www.nist.gov/sha/`, October 2000.

[92] K. Nyberg and R. A. Rueppel. Message Recovery for Signature Schemes Based on the Discrete Logarithm Problem. In Alfredo De Santis, editor, *Advances in Cryptology – EUROCRYPT'94*, volume 950 of *Lecture Notes in Computer Science*, pages 182–193, Perugia, Italy, May 9–12, 1994. Springer-Verlag, Berlin, Germany.

[93] T. Okamoto and D. Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. In David Naccache, editor, *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 159–175, San Francisco, CA, USA, April 8–12, 2001. Springer-Verlag, Berlin, Germany.

[94] T. Okamoto and D. Pointcheval. The Gap-Problems: a New Class of Problems for the Security of Cryptographic Schemes. In Kwangjo Kim, editor, *PKC 2001: 4th International Workshop on Theory and Practice in Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, Cheju Island, South Korea, February 13–15, 2001. Springer-Verlag, Berlin, Germany.

[95] P. Paillier. Public-Key Cryptosystems Based on Discrete Logarithms Residues. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238, Prague, Czech Republic, May 2–6, 1999. Springer-Verlag, Berlin, Germany.

[96] D. H. Phan and D. Pointcheval. Chosen-Ciphertext Security without Redundancy. In Chi-Sung Laih, editor, *Advances in Cryptology – ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 1–18, Taipei, Taiwan, November 30 – December 4, 2003. Springer-Verlag, Berlin, Germany. Full version available from `http://www.di.ens.fr/users/phan/`.

[97] D. H. Phan and D. Pointcheval. About the Security of Ciphers (Semantic Security and Pseudo-Random Permutations). In Helena Handschuh and Anwar Hasan, editors, *SAC 2004: 11th Annual International Workshop on Selected Areas in Cryptography*, volume 3357 of *Lecture Notes in Computer Science*, pages 185–200, Waterloo, Ontario, Canada, August 9–10, 2004. Springer-Verlag, Berlin, Germany. Full version available from `http://www.di.ens.fr/users/phan/`.

[98] D. H. Phan and D. Pointcheval. OAEP 3-Round: A Generic and Secure Asymmetric Encryption Padding. In Pil Joong Lee, editor, *Advances in Cryptology – ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 63–77, Jeju Island, Korea, December 5–9, 2004. Springer-Verlag, Berlin, Germany. Full version available from `http://www.di.ens.fr/users/phan/`.

[99] D. H. Phan and D. Pointcheval. On the Security Notions for Public-Key Encryption Schemes. In Carlo Blundo and Stelvio Cimato, editors, *SCN 04: 4th International Conference on Security in Communication Networks*, Lecture Notes in Computer Science, pages 33–47, Amalfi, Italy, September 8–10, 2005. Springer-Verlag, Berlin, Germany. Full version available from `http://www.di.ens.fr/users/phan/`.

[100] D. Pointcheval. Chosen-Ciphertext Security for any One-Way Cryptosystem. In Hideki Imai and Yuliang Zheng, editors, *PKC 2000: 3rd International Workshop on Theory and Practice in Public Key Cryptography*, volume 1751 of *Lecture Notes in Computer Science*, pages 129–146, Melbourne, Victoria, Australia, January 18–20, 2000. Springer-Verlag, Berlin, Germany.

[101] D. Pointcheval. How to Encrypt Properly with RSA. *CryptoBytes*, 5(1):10–19, winter/spring 2002.

[102] D. Pointcheval. *Le chiffrement asymétrique et la sécurité prouvée.* Thèse d'habilitation, Université de Paris VII, juin 2002.

[103] M. O. Rabin. Digitalized Signatures and Public Key Functions as Intractable as Factoring. Technical Report TR-212, MIT Laboratory for Computer Science, 1979.

[104] C. Rackoff and D. R. Simon. Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444, Santa Barbara, CA, USA, August 11–15, 1992. Springer-Verlag, Berlin, Germany.

[105] R. Rivest. The MD5 Message-Digest Algorithm. RFC 1321, The Internet Engineering Task Force, April 1992.

[106] R. Rivest. All-or-Nothing Encryption and the Package Transform. In *Proceedings of the 4th FSE*, LNCS 1267. Springer-Verlag, Berlin, 1997.

[107] R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public Key Cryptosystems. *Communications of the ACM*, 21(2):120–126, February 1978.

[108] R. Rivest, A. Shamir, and A. Tauman. How to Leak a Secret. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 552–565, Gold Coast, Australia, December 9–13, 2001. Springer-Verlag, Berlin, Germany.

[109] A. Sahai. Non-Malleable Non-Interactive Zero-Knowledge and Chosen-Ciphertext Security. In *Proc. of the 40th FOCS*. IEEE, New York, 1999.

[110] R. Sakai, K. Ohgishi, and M. Kasahara. Cryptosystems Based on Pairing. In *Symposium on Cryptography and Information Security (SCIS2000)*, 2000.

[111] C. P. Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[112] A. Shamir. Identity-Based Cryptosystems and Signature Schemes. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53, Santa Barbara, CA, USA, August 19–23, 1985. Springer-Verlag, Berlin, Germany.

[113] C.E. Shannon. Communication Theory of Secrecy Systems. *Bell Systems Technical Journal*, 28(4):656–715, 1949.

[114] V. Shoup. OAEP Reconsidered. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 239–259, Santa Barbara, CA, USA, August 19–23, 2001. Springer-Verlag, Berlin, Germany.

[115] Victor Shoup. Sequences of Games: a Tool for Taming Complexity in Security Proofs. Cryptology ePrint Archive, Report 2004/332, 2004. `http://eprint.iacr.org/`.

[116] J. Stern, D. Pointcheval, J. Malone-Lee, and N. Smart. Flaws in Applying Proof Methodologies to Signature Schemes. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 93–110, Santa Barbara, CA, USA, August 18–22, 2002. Springer-Verlag, Berlin, Germany.

[117] V.D. To, R.Safavi-Naini, and F. Zhang. New Traitor Tracing Schemes Using Bilinear Map. In *Proceedings of the 2003 ACM Workshop on Digital Rights Management*, pages 67–76, 2003.

[118] E. Verheul. Evidences that XTR is More Secure than Supersingular Elliptic Curve Cryptosystems. In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 195–210, Innsbruck, Austria, May 6–10, 2001. Springer-Verlag, Berlin, Germany.

[119] B. R. Waters. Efficient Identity-Based Encryption Without Random Oracles. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127, Aarhus, Denmark, May 22–26, 2005. Springer-Verlag, Berlin, Germany.

[120] A. C. Yao. Theory and Applications of Trapdoor Functions. In *23rd Annual Symposium on Foundations of Computer Science*, pages 80–91, Chicago, Illinois, November 3–5, 1982. IEEE Computer Society Press.

Résumé

La sécurité prouvée est une branche relativement jeune de la cryptologie dont l'objectif est d'analyser formellement le but ultime des schémas cryptographiques : la sécurité. Elle ne cherche pas à atteindre la sécurité absolue mais plutôt à identifier les conditions suffisantes, au sens de la théorie de la complexité, pour garantir la sécurité. La sécurité prouvée est en étroite relation avec les trois principaux mouvements de la cryptographie : la formalisation des notions de sécurité ; la construction des schémas formellement prouvés sûrs et la recherche de nouvelles fonctionnalités pour la cryptographie.

Dans cette thèse, nous abordons dans un premier temps l'analyse des notions de sécurité, à la fois pour le chiffrement asymétrique et pour le chiffrement symétrique. D'une part, nous étudions en détails les modèles d'attaque et les relations entre eux pour le chiffrement asymétrique. D'autre part, pour le chiffrement par bloc, nous mettons en évidence la relation entre la notion traditionnelle du chiffrement par bloc, *i.e.* permutation (super) pseudo–aléatoire, et avec la notion de base de la confidentialité, *i.e.* sécurité sémantique.

Dans un deuxième temps, nous proposons de nouveaux schémas efficaces et prouvés sûrs pour la cryptographie asymétrique dans le modèle de l'oracle aléatoire (de nouveaux paddings pour le chiffrement et des paddings universels pour le chiffrement et la signature). Nous présentons, de plus, une nouvelle catégorie de schémas prouvés sûrs : les schémas de chiffrement *sans redondance*. Jusqu'à présent, la redondance était nécessaire pour prouver la sécurité.

Nous proposons, dans la troisième partie, une nouvelle fonctionnalité du traçage de pirates pour la diffusion des données chiffrées : la *traçabilité publique*. Nous présentons aussi un schéma satisfaisant cette propriété. Ce schéma est ensuite généralisé à un schéma quasi optimal selon le critère du taux entre de la taille des données chiffrés et celle des données originelles.

Mots-clés : cryptologie, sécurité prouvée, chiffrement à clef publique, modèle de l'oracle aléatoire, padding universel, traçage de traîtres, couplages.

Abstract

Provable security is nowadays one of the major lines of research in Cryptography. It aims at providing security proofs of cryptographic schemes in a complexity–theoretical sense: if one can break the scheme, one can solve the underlying problem. Provable security is strongly related to three main trends in the development of Cryptology: formalization of security notions, design of cryptographic systems, and development of new cryptographic features.

In this thesis, we first deal with notions of security in both asymmetric and symmetric encryption. We study more in detail the relation between different attack models in asymmetric encryption. We also establish the relation between the notion of (super) pseudo–random permutation and that of semantic security in symmetric encryption.

Secondly, we propose new efficient constructions for asymmetric encryption in the random oracle model (new paddings for encryption, and universal paddings for both encryption and signature). Furthermore, we introduce a new class of public–key encryption schemes: chosen ciphertext secure schemes *without redundancy*. Up to now, redundancy used to be required for proofs of security in public–key encryption schemes.

Finally, we consider the traitor tracing problem in broadcast encryption and we introduce a new feature: *public traceability*. We construct a basic scheme with such feature, and then generalize it to achieve almost optimal transmission rates.

Keywords: Cryptology, Provable Security, Public–key Encryption, Random Oracle Model, Universal Padding, Traitor Tracing, Pairings.