Cours de cryptologie avancée

Guilhem Castagnos

Septembre – Décembre 2024

version du 1^{er} septembre 2024

Table des matières

Chapitre I

Bibliographie

Jonathan Katz and Yehuda Lindell *Introduction to Modern Cryptography* 2007 by Chapman & Hall/CRC Press

Oded Goldreich *Foundations of Cryptography - Volume 1 & 2* 2001 and 2004 Cambridge University Press. voir aussi en ligne : `http://www.wisdom.weizmann.ac.il/~oded/foc.html`

Shafi Goldwasser and Mihir Bellare *Lecture Notes on Cryptography* 2008
`http://cseweb.ucsd.edu/~mihir/papers/gb.html`

David Pointcheval *Le Chiffrement Asymétrique et la Sécurité Prouvée* Habilitation à diriger des recherches 2002
`https://www.di.ens.fr/~pointche/Documents/Reports/2002_HDRThesis.pdf`

Victor Shoup *Sequences of Games : A Tool for Taming Complexity in Security Proofs* 2006
`https://www.shoup.net/papers/games.pdf`

Chapitre II

Sécurité CPA

1. Introduction à la sécurité réductionniste

Pour valider la sécurité d'un système cryptographique de façon formelle, Goldwasser et Micali ont proposé en 1984 l'idée de la **sécurité réductionniste** (ou sécurité prouvée) qui tente de lier la sécurité d'un tel système à la complexité d'un problème algorithmique jugé difficile, tel que la factorisation des nombres entiers ou le calcul de logarithmes discrets.

Pour cela on prouve que si un attaquant arrive à casser un schéma cryptographique alors il peut résoudre un problème algorithmique, P_A (calculatoire ou décisionnel). Si P_C désigne le problème de casser le schéma cryptographique, on établit ainsi une réduction algorithmique polynomiale de P_A vers P_C : $P_A \propto P_C$. Ceci assure que si un attaquant est capable de casser le schéma cryptographique en temps polynomial alors il peut aussi résoudre le problème P_A en temps polynomial. On peut être plus précis en donnant les paramètres de la réduction : le temps qu'elle prend, sa probabilité de réussite, le nombre d'appel à un oracle de déchiffrement... On parle parfois de **sécurité exacte**. Ceci permet d'obtenir une **sécurité pratique** du schéma : une sélection des paramètres à utiliser pour obtenir une sécurité donnée.

On fait ensuite l'hypothèse que P_A est un problème difficile, c'est à dire qu'il n'est pas résoluble en temps polynomial. Par contraposée (de la proposition P_C est facile implique P_A est facile) on obtient donc qu'un attaquant polynomial contre le schéma cryptographique ne peut exister. C'est ce que l'on appelle une **preuve de sécurité**.

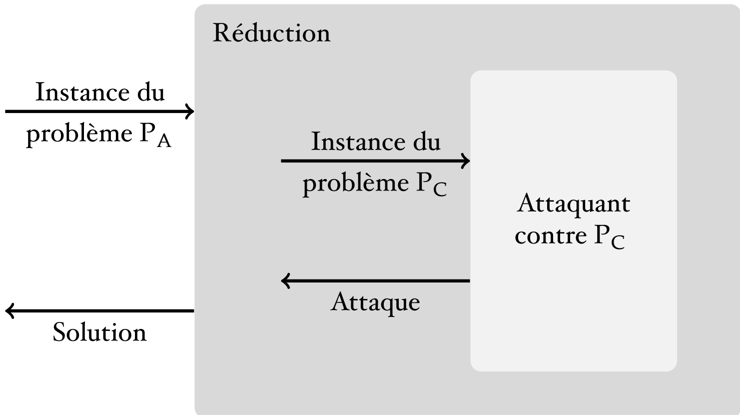


Schéma d'une preuve :

1. Hypothèse algorithmique : P_A est difficile : il ne peut être résolu par un algorithme polynomial à part avec une probabilité négligeable ;

2. Supposer l'existence d'un attaquant efficace $\mathscr{A}$ contre P_C : $\mathscr{A}$ est un algorithme polynomial probabiliste cassant le problème P_C avec bonne probabilité de succès

3. Étant donné une instance de P_A, construire un algorithme polynomial probabiliste $\mathscr{B}$, une réduction, utilisant $\mathscr{A}$ comme sous routine pour résoudre cette instance avec bonne probabilité de succès

4. Conclusion : l'existence de $\mathscr{B}$ contredit l'hypothèse algorithmique. Donc aucun attaquant efficace $\mathscr{A}$ contre P_C ne peut exister. Le schéma cryptographique est donc sûr.

Pour établir de telles preuves, il faut tout d'abord définir un modèle de sécurité : une définition formelle du schéma cryptographique et d'un adversaire contre ce schéma. On établit généralement une hiérarchie de niveaux de sécurité (reposant sur divers problèmes algorithmiques) correspondant à différents buts et moyens pour l'attaquant. Pour le chiffrement asymétrique, un attaquant a inévitablement accès à la clef publique et à l'algorithme de chiffrement, il peut obtenir le chiffrement des messages clairs de son choix. On parle d'attaques à clairs choisis ou *Chosen Plaintext Attack*, noté CPA. Ce moyen d'attaque passif fait l'objet de ce premier chapitre.

2. Fonctions à sens unique et à trappe

Les fonctions à sens unique sont des fonctions qui sont « faciles » à évaluer mais « difficile » à inverser. L'existence de telles fonctions fondamentales pour la cryptographie est une conjecture. Prouver leur existence prouverait que $P \neq NP$. Le fait qu'une fonction à sens unique est « facile » à évaluer se formalise par le fait qu'il existe un algorithme polynomial d'évaluation. Pour l'inversion on requiert qu'un algorithme probabiliste polynomial $\mathscr{A}$ tentant d'inverser la fonction n'a qu'une probabilité extrêmement faible d'y arriver, la probabilité étant prise sur le choix de l'antécédent et sur les choix aléatoires de $\mathscr{A}$. Pour quantifier le « extrêmement faible », on introduit les fonctions négligeables.

Définition II − 1. Une fonction $\nu : \mathbf{N} \to \mathbf{R}^+$ est **négligeable** si pour tout $c \in \mathrm{N}$, il existe un entier k_0 tel que pour tout $k \geqslant k_0$, $\nu(k) \leqslant \frac{1}{k^c}$.

On peut de manière équivalente définir une fonction négligeable comme étant inférieure à l'inverse de tout polynôme positif à partir d'un certain rang. Si un algorithme $\mathscr{A}$ a une probabilité de succès négligeable vue comme fonction de la longueur de son entrée alors l'algorithme construit en répétant $\mathscr{A}$ un nombre polynomial de fois, aura toujours une probabilité de succès négligeable.

Définition II − 2. Une fonction $f : \{0,1\}^* \to \{0,1\}^*$ est à **sens-unique** si

- il existe un algorithme polynomial probabiliste qui sur l'entrée x calcule $f(x)$;
- pour tout algorithme polynomial probabiliste $\mathscr{A}$ il existe une fonction négligeable ν telle que pour k assez grand,

$$\mathbf{Succ}^{\mathsf{OW}}_{f,k}(\mathscr{A}) := \Pr\Big[f(z) = y : x \xleftarrow{\$} \{0,1\}^k; y \leftarrow f(x); z \leftarrow \mathscr{A}(1^k, y)\Big] \leqslant \nu(k)$$

La probabilité est prise sur les choix de x de k bits et les choix faits par $\mathscr{A}$. On fait dépendre $\mathscr{A}$ de 1^k pour permettre d'être polynomial en $|x|$ même si y est beaucoup plus petit que x par exemple $|y| = \log|x|$.

Notations dans la définition : Soit $k \in \mathbf{N}$, 1^k désigne k en notation unaire (1111 … 11 avec k symboles 1). Ainsi la taille de 1^k, $|1^k| = k$. Comme en général on définit la complexité d'un algorithme en fonction de la taille de son entrée, lui donner 1^k en entrée permet d'exprimer sa complexité en fonction de k.

$x \xleftarrow{\$} \mathrm{S}$ signifie que x est tiré avec probabilité uniforme dans S.

Formellement, dans la définition, $x \xleftarrow{\$} \{0,1\}^k$ signifie que x (ou plus rigoureusement x_k) est une (famille de) variable aléatoire discrète suivant la loi uniforme sur $\{0,1\}^k$. Les autres variables y, z sont aussi à considérer comme des (familles de) variables aléatoires discrètes.

L'algorithme polynomial probabiliste $\mathscr{A}$ (l'adversaire), peut être formellement modélisé par une machine de Turing probabiliste. La sortie de $\mathscr{A}$ dépend de son entrée x mais également des choix aléatoires qu'il fait. On peut voir $\mathscr{A}$ comme une fonction déterministe (donc une fonction au sens mathématique) prenant en entrée deux variables aléatoires : x et une chaîne de bits r de longueur suffisamment longue, uniformément distribuée et indépendante de toute autre variable (en particulier de x), rassemblant les choix aléatoires fait par $\mathscr{A}$. La sortie de $\mathscr{A}$ est aussi une variable aléatoire.

Cette définition permet de considérer une sécurité en moyenne.

Exemple : Considérer f la fonction identité et calculer le succès de $\mathscr{A}$ définit ainsi : $\mathscr{A}(1^k, y)$ tire un bit b, si $b = 0$, $\mathscr{A}$ retourne y. Si $b = 1$, $\mathscr{A}$ retourne la chaîne de bit nulle de longueur k : $0 \dots 0$

On peut aussi formuler le succès sous forme d'expérience : on définit l'expérience $\mathbf{Exp}^{\mathsf{OW}}_{f,k}(\mathscr{A})$:

1. Choisir l'entrée $x \overset{\$}{\leftarrow} \{0,1\}^k$, calculer $y := f(x)$
2. $\mathscr{A}$ prend 1^k et y en entrée et renvoie z
3. La sortie de l'expérience est 1 si $f(z) = y$ et 0 sinon

On a alors $\mathbf{Succ}^{\mathsf{OW}}_{f,k}(\mathscr{A}) = \Pr\left[\mathbf{Exp}^{\mathsf{OW}}_{f,k}(\mathscr{A}) = 1\right]$.

Remarques : Dans cette définition, la sécurité est asymptotique pour des tailles d'éléments x assez grand. Pour garantir une sécurité plus pratique on considère des familles de fonctions à sens-unique paramétrées par un paramètre de sécurité. Un exemple est la fonction d'exponentiation $x \to g^x$.

Les fonctions à sens-unique suffisent pour montrer l'existence de plusieurs primitives cryptographiques telles que les générateurs pseudo aléatoires, les *message authentication code*, les schémas de signatures... Pour le chiffrement asymétrique, on a besoin de plus : les fonctions à trappe.

Définition II – 3. Une **fonction à trappe** est une fonction à sens-unique $f : \{0,1\}^* \to \{0,1\}^*$ telle qu'il existe un polynôme p et un algorithme probabiliste polynomial $\mathscr{I}$ tel que pour tout entier k il existe $t_k \in \{0,1\}^*$ tel que $|t_k| \leqslant p(k)$ et pour tout $x \in \{0,1\}^k$, $\mathscr{I}(t_k, f(x)) = y$ tel que $f(y) = f(x)$.

De même on peut définir des familles de fonctions à trappe. Un exemple est la famille de fonctions RSA.

RSA (78) : Soit un nombre RSA n de k bits facteur de deux nombres premiers aléatoires distincts p et q de $k/2$ bits et e un nombre aléatoire positif premier avec $\varphi(n)$ et d l'inverse de e modulo $\varphi(n)$. Pour l'indice (n,e) on associe la trappe d et la fonction RSA $_{(n,e)}$: $(\mathbf{Z}/n\mathbf{Z})^\times \to (\mathbf{Z}/n\mathbf{Z})^\times : x \mapsto x^e \mod n$. C'est en fait une famille de permutations à trappe. Cette fonction est bien évaluable en temps polynomial et inversible en temps polynomial connaissant la trappe, elle est à sens-unique sous l'hypothèse RSA :

Définition II – 4. Soit un algorithme GenRSA qui prend en entrée 1^k et ressort les paramètres n, e, d de RSA. Soit $\mathscr{A}$ un attaquant, on définit l'expérience $\mathbf{Exp}^{\mathsf{OW}}_{\mathsf{GenRSA},k}(\mathscr{A})$:

1. Lancer GenRSA avec entrée 1^k pour obtenir n, e, d
2. Choisir $y \overset{\$}{\leftarrow} (\mathbf{Z}/n\mathbf{Z})^\times$ (équivalent de choisir x aléatoire)
3. $\mathscr{A}$ prend n, e, y en entrée et renvoie $z \in (\mathbf{Z}/n\mathbf{Z})^\times$
4. La sortie de l'expérience est 1 si $z^e = y$ et 0 sinon

On dit que le problème RSA est dur relativement à GenRSA si pour tout algorithme probabiliste polynomial $\mathscr{A}$ il existe une fonction négligeable ν telle que pour k assez grand,

$$\Pr\left[\mathbf{Exp}^{\mathsf{OW}}_{\mathsf{GenRSA},k}(\mathscr{A}) = 1\right] \leqslant \nu(k).$$

L'hypothèse RSA est qu'il existe un générateur GenRSA tel que le problème RSA soit dur.

3. Schéma de chiffrement asymétrique

Définition II – 5 (Schéma de chiffrement asymétrique)**.** Soit $k \in \mathbf{N}$ un paramètre de sécurité. Un schéma de chiffrement asymétrique Π est la donnée d'un triplet de trois algorithmes $(\mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ satisfaisant les trois propriétés suivantes :

1. L'algorithme KeyGen est appelé algorithme de génération de clef. C'est un algorithme probabiliste polynomial qui prend en entrée 1^k et qui retourne un couple (pk, sk) constitué d'une clef publique, pk, et d'une clef secrète, sk. On notera $(pk, sk) \leftarrow \mathsf{KeyGen}(1^k)$

2. L'algorithme Encrypt est appelé algorithme de chiffrement. C'est un algorithme polynomial probabiliste qui prend en entrée une clef publique pk et un message clair, m, élément de $\mathscr{M}$ (l'espace des messages clairs). L'algorithme Encrypt retourne un chiffré, c, élément de $\mathscr{C}$ (l'espace des chiffrés). On notera $c \leftarrow \mathsf{Encrypt}(pk, m)$

3. L'algorithme Decrypt est appelé algorithme de déchiffrement. C'est un algorithme polynomial déterministe qui prend en entrée une clef secrète sk et un chiffré c élément de $\mathscr{C}$ et qui retourne la valeur $\mathscr{D}(sk, c)$. Cette valeur sera soit un message clair élément de $\mathscr{M}$, soit une erreur notée $\perp$ qui indique que le chiffré n'est pas valide.

Le schéma doit être correct, c'est à dire que pour tout paramètre de sécurité k, et pour tout message $m \in \mathscr{M}$, si $(pk, sk) \leftarrow \mathsf{KeyGen}(1^k)$, alors

$$\mathsf{Decrypt}(sk, \mathsf{Encrypt}(pk, m)) = m$$

avec probabilité 1. La probabilité étant sur tous les choix aléatoires que font les algorithmes KeyGen et Encrypt.

Exemples :

1. Le schéma de chiffrement trivial où $\mathsf{Encrypt}(pk, m) = m$ et $\mathsf{Decrypt}(sk, c) = c$: il faut définir des notions de sécurité !

2. Une famille de permutations à trappe permet de définir un système de chiffrement. KeyGen sélectionne une fonction trappe f comme clef publique, sa trappe t_k et l'algorithme $\mathscr{I}$ comme clef privée. À tout message m, $\mathsf{Encrypt}(f, m) = f(m)$. Étant donné c et t_k, $\mathsf{Decrypt}(t_k, c) := \mathscr{I}(t_k, c) = f^{-1}(c)$. Avec la permutation à trappe RSA, cela donne le système ***textbook*** RSA.

La sécurité de base pour un système de chiffrement est le **bris total**. Celui ci est atteint si l'attaquant est capable de retrouver la clef privée au moyen de la clef publique. On note $\mathsf{TB-CPA}$ une telle attaque. Un système sera donc sûr contre le bris total si un attaquant probabiliste polynomial n'a qu'une probabilité négligeable d'y arriver. Pour RSA cela revient à retrouver la trappe d, ce qui est équivalent à factoriser n. RSA est donc $\mathsf{TB-CPA}$ sûr sous l'hypothèse que la factorisation est difficile.

Une sécurité plus forte est la notion de sens-unique :

Définition II – 6 ($\mathsf{OW-CPA}$)**.** Soit $\Pi = (\mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ un schéma de chiffrement asymétrique. Soit $\mathscr{A}$ un algorithme attaquant Π. On définit sa probabilité de succès dans l'inversion ponctuelle du schéma par

$$\mathbf{Succ}_{\Pi}^{\mathsf{OW}}(\mathscr{A}) := \Pr\Big[\mathscr{A}(pk, c) = m : (pk, sk) \leftarrow \mathsf{KeyGen}(1^k); m \xleftarrow{\$} \mathscr{M}; c \leftarrow \mathsf{Encrypt}(pk, m)\Big].$$

On dira que le schéma Π est sûr au sens $\mathsf{OW-CPA}$ si ce succès est négligeable pour tout algorithme polynomial probabiliste $\mathscr{A}$.

Cette sécurité est garantie si on utilise une famille de fonction à trappe comme dans le second exemple. Elle n'est cependant pas suffisante. Le fait que la fonction de chiffrement f utilisée soit à sens-unique ne garantie pas que $f(x)$ cache toutes les informations sur x et l'attaquant peut se contenter d'une information partielle sur le message clair. Ce problème est flagrant si la fonction de chiffrement est déterministe : supposons que $\mathscr{M} = \{0,1\}$ alors un adversaire peut calculer les chiffrés de 0 et de 1 et les reconnaître dans un message chiffré. Ainsi le ***textbook*** RSA vu plus haut n'atteint pas un niveau de sécurité suffisant pour être utilisé tel quel en pratique.

4. Sécurité sémantique

En cryptographie asymétrique, la sécurité inconditionnelle ou parfaite au sens de Shannon n'est pas possible. La notion de **sécurité sémantique** (ou sécurité polynomiale) est qu'un attaquant ne puisse extraire aucune information en temps polynomial sur un message clair à partir de l'un des chiffrés, en dehors de celles qu'il aurait pu obtenir sans ce chiffré (notion introduite par Goldwasser et Micali en 1984).

Définition II – 7 (Sécurité sémantique CPA)**.** Soit $\Pi = (\mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ un schéma de chiffrement asymétrique. Soit $\mathscr{A}$ un algorithme probabiliste polynomial attaquant Π et k un paramètre de sécurité, et soit $\mathscr{E}nv$ un algorithme polynomial probabiliste représentant le contexte. On définit l'expérience de sécurité sémantique CPA, $\mathbf{Exp}_{\Pi,k}^{\mathsf{sem-sec-CPA}}(\mathscr{A})$:

1. On lance l'algorithme $\mathsf{KeyGen}(1^k)$ pour obtenir les clefs (pk, sk)
2. On donne la clef pk à $\mathscr{E}nv$ qui retourne un message m, une relation binaire R et une information publique partielle (donnée par le contexte) α
3. On calcule c un chiffré de m : $c \leftarrow \mathsf{Encrypt}(pk, m)$ le ***challenge*** et on donne (c, α, pk) à $\mathscr{A}$
4. $\mathscr{A}$ retourne une information z
5. La sortie de l'expérience est 1 si z est en relation avec m, c'est à dire si $\mathrm{R}(m, z)$ et 0 sinon.

Soit $\mathscr{S}$ un algorithme probabiliste polynomial, jouant le rôle de simulateur On définit maintenant l'expérience de sécurité sémantique CPA simulée, $\mathbf{Exp}_{\Pi,k}^{\mathsf{sem-sec-simul-CPA}}(\mathscr{S})$:

1. On lance l'algorithme $\mathsf{KeyGen}(1^k)$ pour obtenir les clefs (pk, sk)
2. On donne la clef pk à $\mathscr{E}nv$ qui retourne un message m, une relation binaire R et une information publique partielle α
3. On donne (α, pk) à $\mathscr{S}$
4. $\mathscr{S}$ retourne une information z
5. La sortie de l'expérience est 1 si z est en relation avec m, c'est à dire si $\mathrm{R}(m, z)$ et 0 sinon.

On suppose que les points 1 et 2 des deux expériences sont communs. Le schéma Π est sûr au sens $\mathsf{sem-sec-CPA}$ si pour tous les algorithmes polynomiaux probabilistes $\mathscr{A}$ et $\mathscr{E}nv$, il existe un algorithme probabiliste polynomial $\mathscr{S}$ et une fonction négligeable ν telle que pour k assez grand,

$$\left|\Pr\left(\mathbf{Exp}_{\Pi,k}^{\mathsf{sem-sec-CPA}}(\mathscr{A}) = 1\right) - \Pr\left(\mathbf{Exp}_{\Pi,k}^{\mathsf{sem-sec-simul-CPA}}(\mathscr{S}) = 1\right)\right| \leqslant \nu(k)$$

Dans la définition l'algorithme $\mathscr{E}nv$ permet de préciser le contexte dans lequel l'émetteur, le récepteur et l'adversaire évoluent : un événement force l'émission d'un message chiffré. Le clair correspondant est généré par $\mathscr{E}nv$ ainsi qu'une information publique α, représentant le contexte, que connaîtra l'adversaire. L'algorithme $\mathscr{E}nv$ produit également une relation binaire R : l'adversaire va essayer de deviner à partir du chiffré une information $z \in \mathscr{Z}$ sur le message clair et sera gagnant si z est bien en relation avec m. C'est à dire que $\mathrm{R} \subset \mathscr{M} \times \mathscr{Z}$ et $\mathrm{R}(m,z)$.

Un exemple : Oscar sait qu'Alice et Bob sont dans la même ville aujourd'hui, qu'ils doivent se rencontrer et qu'Alice va envoyer un message chiffré à Bob pour préciser où se rencontrer : le nom de la rue m. Toute cette information publique est représentée par α. Une information z sur m que peut essayer de trouver Oscar est la première lettre du nom de la rue. On dira que m et z sont en relation et qu'Oscar a gagné si z représente la première lettre de m.

Comme dit précédemment, on cherche à quantifier l'information obtenue sur m à partir de c comparé à celle que l'on aurait pu obtenir sans c, à partir du seul contexte. L'expérience simulée mesure la probabilité d'obtenir l'information avec le seul contexte (les 3/4 des rues de la ville commencent par la lettre S...). Ainsi, l'adversaire réussit vraiment son attaque s'il obtient une réelle information, c'est à dire si sa probabilité de succès n'est pas trop proche de celle de la simulation et qu'elle est donc exploitable en pratique.

Dans la pratique, on utilise une notion moins naturelle mais bien plus simple à manipuler, la notion d'**indistinguabilité** : si m_0 et m_1 sont deux messages clairs et si c est un chiffré de l'un de ces deux messages, alors un attaquant ne peut pas décider si c a été retourné par $\mathsf{Encrypt}(pk, m_0)$ ou par $\mathsf{Encrypt}(pk, m_1)$, toujours en temps polynomial. Cette notion, introduite aussi par Goldwasser et Micali est équivalente à la notion de sécurité sémantique (voir plus loin).

Définition II − 8 ($\mathsf{IND-CPA}$)**.** Soit $\Pi = (\mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ un schéma de chiffrement asymétrique. Soit $\mathscr{A} = (\mathscr{A}_1, \mathscr{A}_2)$ un algorithme attaquant Π et k un paramètre de sécurité. On définit l'expérience d'indistinguabilité CPA, $\mathbf{Exp}_{\Pi,k}^{\mathsf{IND-CPA}}(\mathscr{A})$:

1. On lance l'algorithme $\mathsf{KeyGen}(1^k)$ pour obtenir les clefs (pk, sk)
2. $\mathscr{A}_1$ reçoit pk et retourne (m_0, m_1, s) avec deux messages de $\mathscr{M}$ de même longueur et un état d'information s
3. On choisit un bit aléatoire $b^\star \xleftarrow{\$} \{0,1\}$
4. On calcule $c^\star$ un chiffré de $m_{b^\star}$: $c^\star \leftarrow \mathsf{Encrypt}(pk, m_{b^\star})$: c'est le *challenge* et on donne $(s, c^\star)$ à $\mathscr{A}_2$
5. $\mathscr{A}_2$ sort un bit b
6. La sortie de l'expérience est 1 si $b = b^\star$ et 0 sinon.

L'avantage de l'attaquant $\mathscr{A}$ pour résoudre l'indistinguabilité du schéma Π est défini par

$$\mathbf{Adv}_{\Pi,k}^{\mathsf{IND-CPA}}(\mathscr{A}) = \left|\Pr\left(\mathbf{Exp}_{\Pi,k}^{\mathsf{IND-CPA}}(\mathscr{A}) = 1\right) - \frac{1}{2}\right|.$$

Le schéma Π est sûr au sens $\mathsf{IND-CPA}$ si pour tout algorithme polynomial probabiliste $\mathscr{A}$ il existe une fonction négligeable ν telle que pour k assez grand,

$$\mathbf{Adv}_{\Pi,k}^{\mathsf{IND-CPA}}(\mathscr{A}) \leqslant \nu(k)$$

Remarques

1. L'état d'infomation s permet de « transmettre » l'état de $\mathscr{A}_1$ à $\mathscr{A}_2$. En particulier, on peut supposer que s contient toujours les messages m_0 et m_1.
2. Dans la définition précédente, le terme avantage signifie l'avantage de l'algorithme $\mathscr{A}$ par rapport à un lancer de pièce : si $\mathscr{A}_2$ tire au hasard sa réponse, on aura $\mathbf{Adv}_{\Pi,k}^{\mathsf{IND-CPA}}(\mathscr{A}) = 0$.
3. Soit Π un schéma de chiffrement, si l'algorithme $\mathsf{Encrypt}$ est déterministe alors Π n'est pas sûr au sens $\mathsf{IND-CPA}$.

4. On définit parfois l'avantage de la manière suivante :

$$\mathbf{Adv}_{\Pi,k}^{\mathsf{IND-CPA}'}(\mathscr{A}) = \left| 2\Pr\left(\mathbf{Exp}_{\Pi,k}^{\mathsf{IND-CPA}}(\mathscr{A}) = 1\right) - 1 \right|.$$

soit deux fois l'avantage précédent (cela ne change pas grand chose pour des considérations asymptotiques). En utilisant les formules suivantes (on allège les notations)

$$\begin{aligned}\Pr\left[b = b^\star\right] &= \Pr\left[b = b^\star | b^\star = 1\right] \times \Pr\left[b^\star = 1\right] + \Pr\left[b = b^\star | b^\star = 0\right] \times \Pr\left[b^\star = 0\right] \\ &= \frac{\Pr\left[b = 1 | b^\star = 1\right]}{2} + \frac{\Pr\left[b = 0 | b^\star = 0\right]}{2},\end{aligned}$$

et

$$\Pr\left[b = 1 | b^\star = 0\right] + \Pr\left[b = 0 | b^\star = 0\right] = 1,$$

on voit facilement qu'avec cette deuxième notation de l'avantage on a

$$\begin{aligned}\mathbf{Adv}_{\Pi,k}^{\mathsf{IND-CPA}'}(\mathscr{A}) &= \left| \Pr\left[b = 1 | b^\star = 1\right] + \Pr\left[b = 0 | b^\star = 0\right] - 1 \right| \\ &= \left| \Pr\left[b = 1 | b^\star = 1\right] - \Pr\left[b = 1 | b^\star = 0\right] \right| \\ &= \left| \Pr\left[b = 0 | b^\star = 1\right] - \Pr\left[b = 0 | b^\star = 0\right] \right|\end{aligned}$$

C'est à dire l'avantage de $\mathscr{A}$ pour distinguer entre les deux distributions : les chiffrés tel que $b^\star = 1$ et ceux tel que $b^\star = 0$.

Théorème II – 9 (Goldwasser - Micali (84)). Un schéma de chiffrement asymétrique est sémantiquement sûr CPA si et seulement si il est IND – CPA sûr.

Démonstration. On suit une preuve de Dodis.

On commence par montrer que IND – CPA $\Rightarrow$ sémantiquement sûr CPA.

Soit un schéma Π sûr au sens IND – CPA. On suppose qu'il n'est pas sémantique sûr, c'est à dire qu'il existe un adversaire $\mathscr{A}$ contre cette notion. On va construire un adversaire $\mathscr{B}$ contre l'IND – CPA. Si cet adversaire est efficace, on obtiendra une contradiction donc Π est sémantiquement sûr.

Il existe donc deux algorithmes probabilistes polynomiaux $\mathscr{E}nv$ et $\mathscr{A}$ tel que pour tout algorithme probabiliste polynomial simulateur $\mathscr{S}$,

$$\left| \Pr\left(\mathbf{Exp}_{\Pi,k}^{\mathsf{sem-sec-CPA}}(\mathscr{A}) = 1\right) - \Pr\left(\mathbf{Exp}_{\Pi,k}^{\mathsf{sem-sec-simul-CPA}}(\mathscr{S}) = 1\right) \right| \geqslant \epsilon$$

où ϵ est non négligeable. En particulier on peut prendre $\mathscr{S}$ comme suit :

1. $\mathscr{S}$ reçoit (α, pk)
2. Soit c' un chiffré de m' : $c' \leftarrow \mathsf{Encrypt}(pk, m')$ où m' est un message fixe de $\mathscr{M}$ de même longueur que le message m choisit par $\mathscr{E}nv$ (par exemple le message avec tous les bits à 0)
3. $\mathscr{S}$ renvoie $z' \leftarrow \mathscr{A}(c', \alpha, pk)$

Pour résumer on a

1. $(pk, sk) \leftarrow \mathsf{KeyGen}(1^k)$
2. $(m, \mathrm{R}, \alpha) \leftarrow \mathscr{E}nv(pk)$
3. $c \leftarrow \mathsf{Encrypt}(pk, m)$ et $c' \leftarrow \mathsf{Encrypt}(pk, m')$

4. $z \leftarrow \mathscr{A}(c, \alpha, pk)$ et $z' \leftarrow \mathscr{A}(c', \alpha, pk)$

et par hypothèse,

$$|\Pr(\mathrm{R}(m, z)) - \Pr(\mathrm{R}(m, z'))| \geqslant \epsilon.$$

L'algorithme $\mathscr{A}$ va nous permettre de distinguer les chiffrés de m et ceux de m'. On construit maintenant un attaquant probabiliste polynomial $\mathscr{B} = (\mathscr{B}_1, \mathscr{B}_2)$ contre $\mathsf{IND-CPA}$ à partir de $\mathscr{A}$.

On définit $\mathscr{B}_1$:

1. $\mathscr{B}_1$ reçoit pk
2. Il obtient (m, R, α) de $\mathscr{E}nv(pk)$
3. Il ressort (m, m', s) avec $s = (\mathrm{R}, \alpha, pk)$

On définit maintenant $\mathscr{B}_2$:

1. $\mathscr{B}_2$ reçoit $(s, c^\star)$ avec $s = (\mathrm{R}, \alpha, pk)$
2. On obtient $z^\star$ de $\mathscr{A}(c^\star, \alpha, pk)$
3. Si $\mathrm{R}(m, z^\star)$, $\mathscr{B}_2$ sort $b = 0$ (le message est m), sinon $\mathscr{B}_2$ sort $b = 1$ (le message est m').

On note $b^\star$ le bit tiré lors de l'expérience d'indistinguabilité. On a $b^\star = 0$ si $c^\star$ est un chiffré de m et $b^\star = 1$ si c'est un chiffré de m'.

Si $b^\star = 0$, le message est m, $\mathscr{A}$ ressort $z^\star = z$ et $\mathscr{B}_2$ ressort aussi $b = 0$ si $\mathrm{R}(m, z)$ ce qui arrive avec $\Pr(\mathrm{R}(m, z)) = \Pr(b = b^\star | b^\star = 0)$.

Si $b^\star = 1$, le message est m', $\mathscr{A}$ ressort $z^\star = z'$ et $\mathscr{B}_2$ ressort aussi $b = 1$ si $\neg\mathrm{R}(m, z')$ ce qui arrive avec $1 - \Pr(\mathrm{R}(m, z')) = \Pr(b = b^\star | b^\star = 1)$. Au final, la probabilité que l'expérience d'indistinguabilité soit un succès est

$$\Pr(b = b^\star) = \frac{1}{2}\Pr(\mathrm{R}(m, z)) + \frac{1}{2}(1 - \Pr(\mathrm{R}(m, z'))),$$

ce qui donne $\frac{1}{2} + \frac{\Pr(\mathrm{R}(m,z)) - \Pr(\mathrm{R}(m,z'))}{2}$. Donc l'avantage

$$\mathbf{Adv}^{\mathsf{IND-CPA}}_{\Pi,k}(\mathscr{B}) = \left| \frac{\Pr\left(\mathbf{Exp}^{\mathsf{sem-sec-CPA}}_{\Pi,k}(\mathscr{A}) = 1\right) - \Pr\left(\mathbf{Exp}^{\mathsf{sem-sec-simul-CPA}}_{\Pi,k}(\mathscr{S}) = 1\right)}{2} \right|$$

ce qui est non négligeable.

Réciproque : on montre que sémantiquement sûr $\mathsf{CPA} \Rightarrow \mathsf{IND-CPA}$.

On suppose donc que Π est sémantiquement sûr CPA et qu'il n'est pas $\mathsf{IND-CPA}$. Soit $\mathscr{B} = (\mathscr{B}_1, \mathscr{B}_2)$ un attaquant $\mathsf{IND-CPA}$ ayant un avantage non négligeable. On peut supposer que les messages m_0, m_1 choisis par $\mathscr{B}_1$ sont distincts. Cet attaquant peut être vu comme un attaquant contre la sécurité sémantique qui étant donné le contexte suivant « Alice va envoyer un chiffré de m_0 ou de m_1 », obtient, à partir du chiffré, l'information « être un chiffré de m_0 ou de m_1 ». C'est à dire satisfaire une relation d'égalité. Plus précisément, on construit $\mathscr{E}nv$ comme suit :

1. $\mathscr{E}nv$ reçoit pk
2. On donne pk à $\mathscr{B}_1$ qui retourne (m_0, m_1, s) : $(m_0, m_1, s) \leftarrow \mathscr{B}_1(pk)$
3. On pose $\alpha = (m_0, m_1, s)$ et R la relation $\mathrm{R}(m, z) \Leftrightarrow m = z$.
4. On tire $b^\star \xleftarrow{\$} \{0, 1\}$, et on renvoie $(m_{b^\star}, \mathrm{R}, \alpha)$

On définit ensuite l'attaquant $\mathscr{A}$ contre la sécurité sémantique :

1. $\mathscr{A}$ reçoit $(c^\star, \alpha, pk)$ avec $\alpha = (m_0, m_1, s)$

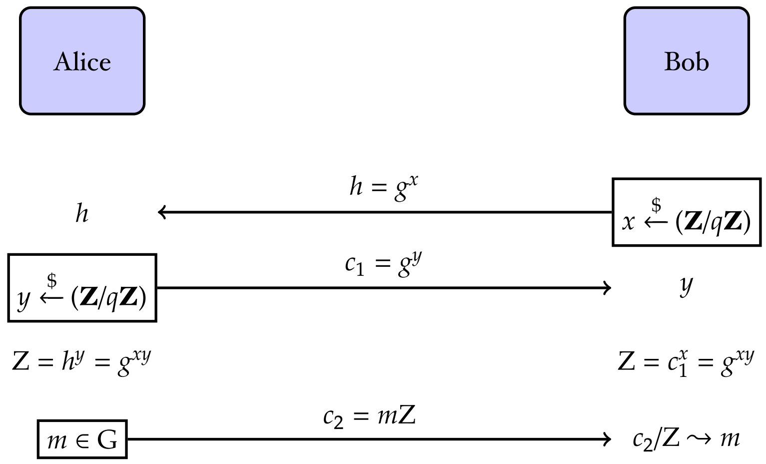
2. On donne $(s, c^\star)$ à $\mathscr{B}_2$ qui retourne un bit b
3. On retourne $z := m_b$

Il est clair que la probabilité $\Pr\left(\mathbf{Exp}^{\mathsf{sem-sec-CPA}}_{\Pi,k}(\mathscr{A}) = 1\right) = \Pr\left(\mathbf{Exp}^{\mathsf{IND-CPA}}_{\Pi,k}(\mathscr{B}) = 1\right)$. Par contre tout algorithme $\mathscr{S}$ qui n'aurait accès qu'à (α, pk) n'aurait aucune information sur le bit $b^\star$ tiré par $\mathscr{E}nv$ ($b^\star$ est déterminé après α). Cet algorithme $\mathscr{S}$ n'a donc qu'une chance sur deux de retourner le bon message. Ainsi, $\Pr\left(\mathbf{Exp}^{\mathsf{sem-sec-simul-CPA}}_{\Pi,k}(\mathscr{S}) = 1\right) = \frac{1}{2}$. Au final, on a

$$\left|\Pr\left(\mathbf{Exp}^{\mathsf{sem-sec-CPA}}_{\Pi,k}(\mathscr{A}) = 1\right) - \Pr\left(\mathbf{Exp}^{\mathsf{sem-sec-simul-CPA}}_{\Pi,k}(\mathscr{S}) = 1\right)\right| =$$
$$\left|\Pr\left(\mathbf{Exp}^{\mathsf{IND-CPA}}_{\Pi,k}(\mathscr{B}) = 1\right) - \frac{1}{2}\right| = \mathbf{Adv}^{\mathsf{IND-CPA}}_{\Pi,k}(\mathscr{B}),$$

qui est non négligeable. □

Exemple Elgamal (85) : Soit GenDH un algorithme polynomial qui prend en entrée 1^k et retourne la description d'un groupe cyclique $\mathbf{G}$ son ordre q (premier avec $|q| = k$ et un générateur g. On suppose que l'on peut calculer dans G en temps polynomial. L'algorithme KeyGen appelle GenDH puis choisit x aléatoire dans $\mathbf{Z}/q\mathbf{Z}$ et calcule $h = g^x$. KeyGen retourne $pk = (\mathbf{G}, q, g, h)$ et $sk = (\mathrm{G}, q, g, x)$. On pose $\mathscr{M} = \mathrm{G}$ et $\mathscr{C} = \mathrm{G} \times \mathrm{G}$. L'algorithme Encrypt sur l'entrée (pk, m) choisit y aléatoirement dans $\mathbf{Z}/q\mathbf{Z}$ et retourne $c = (g^y, mh^y)$. L'algorithme Decrypt sur l'entrée $(sk, (c_1, c_2))$ retourne c_2/c_1^x. On peut voir la deuxième coordonnée d'un chiffré Elgamal $m \mapsto mh^y$ comme un *one-time pad* dans $\mathbf{G}$. Si y est choisi de manière uniforme mh^y n'apporte aucune information sur m, on a donc un chiffrement parfait. Mais comme g^y et h sont aussi donnés, ce n'est pas le cas. En fait, h^y peut être vu comme une clef secrète partagée par un échange de clef Diffie-Hellman ne servant qu'une fois :



où Z est une clef secrète établie par l'échange de clef Diffie-Hellman, utilisée ensuite pour chiffrer m par un *one time pad*.

On définit les problèmes CDH et DDH sur lesquels reposent la sécurité d'Elgamal.

Définition II – 10. Soit $\mathscr{A}$ un attaquant, on définit l'expérience $\mathbf{Exp}^{\mathsf{CDH}}_{\mathsf{GenDH},k}(\mathscr{A})$:

1. Lancer GenDH avec entrée 1^k pour obtenir G, q, g
2. Choisir $x, y \xleftarrow{\$} (\mathbf{Z}/q\mathbf{Z})$, et calculer $\mathrm{X} = g^x$ et $\mathrm{Y} = g^y$
3. $\mathscr{A}$ prend $\mathrm{G}, q, g, (\mathrm{X}, \mathrm{Y})$ en entrée et renvoie $\mathrm{Z} \in \mathrm{G}$

4. La sortie de l'expérience est 1 si $Z = g^{xy}$ et 0 sinon

On dit que le problème CDH est dur relativement à GenDH si pour tout algorithme probabiliste polynomial $\mathscr{A}$ il existe une fonction négligeable ν telle que pour k assez grand, le succès

$$\Pr\left[\mathbf{Exp}^{\mathsf{CDH}}_{\mathsf{GenDH},k}(\mathscr{A}) = 1\right] \leqslant \nu(k).$$

L'hypothèse CDH (*computational Diffie–Hellman*) est qu'il existe un générateur GenDH tel que le problème CDH soit dur.

Définition II – 11. Soit $\mathscr{D}$ un attaquant, on définit l'expérience $\mathbf{Exp}^{\mathsf{DDH}}_{\mathsf{GenDH},k}(\mathscr{D})$:

1. Lancer GenDH avec entrée 1^k pour obtenir G, q, g
2. Choisir $x, y \overset{\$}{\leftarrow} (\mathbf{Z}/q\mathbf{Z})$, et calculer $X := g^x$ et $Y := g^y$ et prendre $b^\star \overset{\$}{\leftarrow} \{0,1\}$
3. Si $b^\star = 0$ alors $Z \overset{\$}{\leftarrow} G$, sinon $Z := g^{xy}$
4. $\mathscr{D}$ prend $G, q, g, (X, Y, Z)$ en entrée et renvoie un bit b
5. La sortie de l'expérience est 1 si $b = b^\star$ et 0 sinon

On dit que le problème DDH est dur relativement à GenDH si pour tout algorithme probabiliste polynomial $\mathscr{D}$ il existe une fonction négligeable ν telle que pour k assez grand,

$$\mathbf{Adv}^{\mathsf{DDH}}_{\mathsf{GenDH},k}(\mathscr{D}) = \left|\Pr\left(\mathbf{Exp}^{\mathsf{DDH}}_{\mathsf{GenDH},k}(\mathscr{D}) = 1\right) - \frac{1}{2}\right| \leqslant \nu(k).$$

L'hypothèse DDH (*Decision Diffie–Hellman*) est qu'il existe un générateur GenDH tel que le problème DDH soit dur.

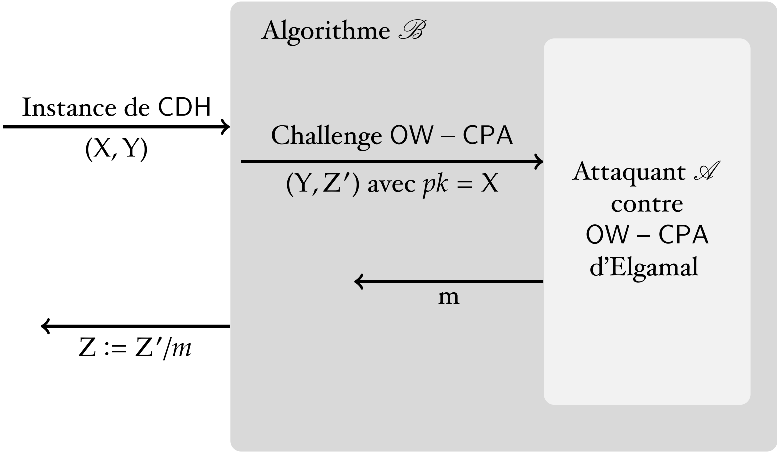
On conjecture que cette hypothèse DDH est vraie dans les sous groupes de $(\mathbf{Z}/p\mathbf{Z})^\times$ d'ordre q avec q un grand diviseur premier de $p-1$ avec p premier (par exemple en utilisant des nombres premiers de Sophie Germain : $p = 2q+1$, on travaille dans ce cas avec les carrés de $(\mathbf{Z}/p\mathbf{Z})^\times$. On utilise aussi certaines courbes elliptiques (en particulier, il ne faut pas qu'un couplage puisse être calculé : étant donné P, aP, bP, cP, on a $e(P, cP) = e(aP, bP)$ si et seulement si $c = ab$ modulo l'ordre de P).

Théorème II – 12. Elgamal est OW – CPA sous l'hypothèse CDH et IND – CPA sous l'hypothèse DDH.

Démonstration. On commence par la sécurité OW–CPA. On suppose donc que Elgamal n'est pas OW–CPA et on construit un algorithme résolvant le problème CDH.

On se donne G, q, g retournés par GenDH. On note $X = g^x$ et $Y = g^y$ le début du triplet Diffie-Hellman à reconstituer. On prend $pk = (\mathbf{G}, q, g, X)$ comme clef publique pour un chiffrement Elgamal : X étant un élément de G tiré uniformément, on obtient bien la même distribution que la clef publique. On construit ensuite le chiffré (Y, Z') où Z' est pris aléatoirement dans G. C'est bien un chiffré valide d'un message aléatoire : c'est évident pour la première coordonnée, et pour la deuxième coordonnée, prendre un message $m \overset{\$}{\leftarrow} G$ et calculer $Z' = mX^y$ revient à prendre directement $Z' \overset{\$}{\leftarrow} G$.

Comme Elgamal est supposé non OW – CPA, il existe un attaquant $\mathscr{A}$ retrouvant le message m dont (Y, Z') est un chiffré avec un succès non négligeable. Mais on aura alors $m = Z'/Y^x$, donc $Z := Z'/m = Y^x = g^{xy}$. On a donc construit un algorithme $\mathscr{B}$ résolvant CDH avec le même succés non-négligeable. On a donc une contradiction. On résume la construction de $\mathscr{B}$ par la figure suivante (bien noter que la vue de $\mathscr{A}$ est bien un challenge OW – CPA pour Elgamal, avec la bonne distribution).



On montre maintenant la sécurité IND – CPA. On procède toujours par l'absurde. Soit $\mathscr{A} = (\mathscr{A}_1, \mathscr{A}_2)$ un adversaire IND – CPA avec un avantage non négligeable. Soit (X, Y, Z) un challenge DDH. On construit un algorithme $\mathscr{D}$ comme suit :

1. Entrée : $G, q, g, (X, Y, Z)$
2. On pose $pk = (\mathbf{G}, q, g, X)$ comme clef publique
3. On obtient m_0, m_1, s par $\mathscr{A}_1(pk)$
4. On tire un bit $b^\star$
5. On construit le chiffré $c^\star = (Y, m_{b^\star} Z)$
6. On donne $c^\star$ avec s à $\mathscr{A}_2$ qui retourne un bit b
7. On retourne 1 si $b = b^\star$ et 0 sinon

Dans le cas où (X, Y, Z) n'est pas un triplet Diffie-Hellman, comme Z est uniformément distribuée dans G indépendante des autres variables aléatoires, la seconde composante du chiffré $c^\star$, $m_{b^\star} Z$ est aussi uniformément distribuée dans G et est indépendante de $m_{b^\star}$ (*one-time pad*). Le chiffré $c^\star$, en particulier, et toutes les entrées de l'adversaire, en général, sont donc indépendantes du bit $b^\star$. Même un adversaire avec une puissance de calcul infini ne peut donc deviner le bit $b^\star$ qu'avec un avantage nul :

$$\Pr[b = b^\star] = \Pr[b = 0 | b^\star = 0] \Pr[b^\star = 0] + \Pr[b = 1 | b^\star = 1] \Pr[b^\star = 1] = \frac{1}{2}(\Pr[b = 0] + \Pr[b = 1]) = \frac{1}{2}.$$

Ainsi $\mathscr{D}$ retourne 0 ou 1 avec probabilité 1/2.

Maintenant si (X, Y, Z) est un triplet valide, $c^\star$ est un chiffré valide de $m^\star$. La vue de $\mathscr{A}$ est donc la même que dans l'expérience d'indistinguabilité. Le distingueur $\mathscr{D}$ retourne donc 1 avec probabilité

$$\Pr\left(\mathbf{Exp}^{\mathsf{IND-CPA}}_{\mathsf{Elgamal},k}(\mathscr{A}) = 1\right).$$

Au final, si b' désigne le bit choisi dans l'expérience DDH, l'expérience retourne 1 sachant que $b' = 0$ avec probabilité 1/2, et retourne 1 sachant que $b' = 1$ avec probabilité $\Pr\left(\mathbf{Exp}^{\mathsf{IND-CPA}}_{\mathsf{Elgamal},k}(\mathscr{A}) = 1\right)$.

On a donc

$$\Pr\left(\mathbf{Exp}^{\mathsf{DDH}}_{\mathsf{GenDH},k}(\mathscr{D}) = 1\right) = \frac{1}{2} \times \frac{1}{2} + \Pr\left(\mathbf{Exp}^{\mathsf{IND-CPA}}_{\mathsf{Elgamal},k}(\mathscr{A}) = 1\right) \times \frac{1}{2}.$$

$$\mathbf{Adv}^{\mathsf{DDH}}_{\mathsf{GenDH},k}(\mathscr{D}) = \frac{1}{2}\left|\Pr\left(\mathbf{Exp}^{\mathsf{IND-CPA}}_{\mathsf{Elgamal},k}(\mathscr{A}) = 1\right) - \frac{1}{2}\right|$$

donc

$$\mathbf{Adv}^{\mathsf{DDH}}_{\mathsf{GenDH},k}(\mathscr{D}) = \frac{1}{2}\mathbf{Adv}^{\mathsf{IND-CPA}}_{\mathsf{Elgamal},k}(\mathscr{A})$$

qui est donc non négligeable.

□

5. Les preuves par jeux

Les preuves de sécurité que nous avons vu jusqu'à présent sont relativement simples. Cependant lorsque l'on considère des protocoles cryptographiques plus avancés que le chiffrement ou des notions de sécurité plus fortes, les preuves s'allongent et il devient difficile pour le lecteur de vérifier que tous les arguments ont bien été énoncé pour le calcul final d'avantage. Aussi, Victor Shoup a formalisé en 2006 le concept des preuves par jeux, permettant une présentation uniforme des preuves de sécurité et de faire apparaître clairement les arguments les uns après les autres.

L'idée est la suivante. On considère d'abord un jeu 0 correspondant à l'expérience définissant la sécurité (par exemple l'expérience $\mathsf{IND-CPA}$). À ce jeu on associe un évènement S_0 dont la probabilité intervient dans la définition du modèle de sécurité (par exemple $b = b^\star$ pour l'$\mathsf{IND-CPA}$). On modifie ensuite de manière minimale ce jeu pour construire des jeux $1, 2, \ldots, n$. On note S_i l'évènement défini au jeu i, avec une définition proche de celle de S_0. L'idée est que $\Pr(S_n)$ soit égale à une probabilité cible (par exemple $1/2$ pour l'$\mathsf{IND-CPA}$) et que $|\Pr(S_i) - \Pr(S_{i+1})|$ soit négligeable pour $i = 0, \ldots, n-1$. Ainsi par inégalité triangulaire, on aura $|\Pr(S_0) - \Pr(S_n)|$ négligeable et le résultat de sécurité.

Pour passer d'un jeu à un autre, on suit en général trois types de transitions :

1. Des transitions consistant à une réécriture équivalente des variables (par exemple $Z = g^{xy}$ au lieu de $Z = c_1^x$ et $c_1 = g^y$), dans ce cas là, $\Pr(S_i) = \Pr(S_{i+1})$.
2. Des transitions basées sur deux distributions D_1, D_2 indistinguables (par exemple (g^x, g^y, g^{xy}) et (g^x, g^y, g^z)) : le jeu i va utiliser des variables de D_1 et le jeu $i+1$, des variables de D_2. Pour prouver que $|\Pr(S_i) - \Pr(S_{i+1})|$ est négligeable, on va prouver qu'il existe un algorithme qui sort 1 avec probabilité $\Pr(S_i)$ (resp. $\Pr(S_{i+1})$) étant donné un élément tiré selon la distribution D_1 (resp. D_2).
3. Une transition basée sur l'occurrence d'un évènement « d'échec », F : les deux jeux procèdent similairement à moins que l'évènement F arrive, c'est à dire que $S_i \wedge \neg F$ est équivalent à $S_{i+1} \wedge \neg F$. On voit alors facilement que $|\Pr(S_i) - \Pr(S_{i+1})| \leqslant \Pr(F)$:

$$\begin{array}{rcl} |\Pr(S_i) - \Pr(S_{i+1})| & = & |\Pr(S_i \wedge F) + \Pr(S_i \wedge \neg F) - \Pr(S_{i+1} \wedge F) - \Pr(S_{i+1} \wedge \neg F)| \\ & = & |\Pr(S_i \wedge F) - \Pr(S_{i+1} \wedge F)| \\ & \leqslant & \Pr(F) \end{array}$$

où la dernière égalité vient du fait que les deux probabilités $\Pr(S_i \wedge F)$ et $\Pr(S_{i+1} \wedge F)$ sont comprises entre 0 et $\Pr(F)$.

Comme exemple, voyons comment réécrire la preuve $\mathsf{IND-CPA}$ d'Elgamal dans ce formalisme, on va essentiellement retrouver notre preuve précédente, mais présentée différemment.

On commence par écrire un jeu 0 qui est l'expérience $\mathsf{IND-CPA}$ d'Elgamal et on lui associe l'évènement $S_0 : b = b^\star$. Ainsi

$$\Pr(S_0) = \Pr\Big(\mathbf{Exp}^{\mathsf{IND-CPA}}_{\Pi,k}(\mathscr{A}) = 1\Big).$$

Ensuite on écrit un jeu 1, similaire au jeu 0 mais on modifiant le calcul du chiffré challenge $c^\star$. Au lieu de calculer $c_2^\star = m_{b^\star} h^y$ où $h = g^x$ est la clef publique, comme dans le jeu 0, on calcule $c_2^\star = m_{b^\star} Z$ avec Z uniforme dans G. On montre ensuite que $Pr(S_1) = 1/2$ car dans le jeu 1, l'adversaire n'a plus d'information sur $b^\star$.

Puis on étudie la différence $|\Pr(S_0) - \Pr(S_1)|$. Au jeu 0, $h = g^x, c_1 = g^y, h^y$ est un triplet Diffie-Hellman, alors qu'au jeu 1, $h = g^x, c_1 = g^y, Z$ est un triplet aléatoire de G^3. On peut alors construire un distingueur prenant en entrée un tel triplet et dont l'exécution correspond au jeu 0 (resp. au jeu 1) si c'est un triplet Diffie-Hellman (resp. un triplet aléatoire). C'est exactement le distingueur vu dans la preuve $\mathsf{IND-CPA}$ d'Elgamal. On a alors $\Pr(S_0)$ (resp. $\Pr(S_1)$) égale à la probabilité que le distingueur retourne 1 étant donné un triplet Diffie-Hellman (resp. un triplet aléatoire). On voit facilement que $|\Pr(S_0)-\Pr(S_1)| = 2\mathbf{Adv}^{\mathsf{DDH}}_{\mathsf{GenDH},k}(\mathscr{D})$, ce qui permet de conclure que

$$\mathbf{Adv}^{\mathsf{IND-CPA}}_{\mathsf{Elgamal},k}(\mathscr{A}) = |\Pr(S_0) - \Pr(S_1)|,$$

est négligeable.

6. Non malléabilité

Le chiffrement Elgamal est malléable : si (c_1, c_2) est un chiffré de la forme (g^y, mh^y) alors, $(c_1, c_2) \times (g^r, m'h^r)$ est un chiffré de mm'. On dit qu'Elgamal est **homomorphe multiplicatif**. Cette propriété peut-être recherché pour des applications comme on le verra dans la section suivante. Cela peut aussi être considéré comme une faiblesse, un attaquant peut ainsi mettre à mal l'intégrité du message clair en transformant (c_1, c_2) en un chiffrement d'un autre message. Ceci est d'autant plus nuisible dans le cas d'attaque active. Une autre notion de sécurité est donc la non malléabilité (NM) introduite par Dolev, Dwork et Naor (91).

De manière informelle, on ne veut pas qu'un attaquant étant donné un chiffré c de m puisse produire un nouveau chiffré c' d'un message m' telle que m et m' soient en relation. On peut formaliser cette notion avec un attaquant à deux étapes comme pour l'indistinguabilité. On montre alors que $\mathsf{NM-IND} \Rightarrow \mathsf{IND-CPA}$. Dans le cas d'attaque actives adaptatives, on a une équivalence entre les deux notions.

7. Résumé

On a

$$\begin{array}{ccccccc} \mathsf{TB-CPA} & \Longleftarrow & \mathsf{OW-CPA} & \Longleftarrow & \mathsf{IND-CPA} & \Longleftarrow & \mathsf{NM-CPA} \\ & & & & \Updownarrow & & \\ & & & & \mathsf{sem-sec-CPA} & & \end{array}$$

On a des séparations pour chaque notion de sécurité de la première ligne, ainsi on peut construire des schémas de chiffrements :

- $\mathsf{TB-CPA}$ mais non $\mathsf{OW-CPA}$: $sk = \$$, $pk = \$$ et $c = m$
- $\mathsf{OW-CPA}$ mais non $\mathsf{IND-CPA}$: *textbook* RSA
- $\mathsf{IND-CPA}$ mais non $\mathsf{NM-CPA}$: Elgamal ou Paillier