Thèse de Doctorat
Université de Caen

Spécialité : Informatique

Les Preuves de Connaissance et leurs Preuves de Sécurité

Soutenue le 12 décembre 1996

David Pointcheval

Travaux effectués au
Laboratoire d'Informatique
École Normale Supérieure

Mihir Bellare *(rapporteur)*
Claude Carlet
Robert Cori
Philippe Flajolet
Marc Girault
Guy Robin *(rapporteur)*
Claus Schnorr *(rapporteur)*
Jacques Stern
Brigitte Vallée *(directeur)*

Résumé

La cryptographie a pour but de garantir des communications sûres, par l'intermédiaire notamment de protocoles de chiffrement et d'authentification. Mais encore faut-il que les schémas proposés satisfassent les propriétés de sécurité énoncées face à n'importe quel attaquant. Cette thèse se propose donc de présenter des schémas cryptographiques prouvés sûrs dans le modèle de l'oracle aléatoire. Ce modèle, formalisé par Bellare et Rogaway en 1993, a ouvert la voie à des preuves formelles de sécurité de schémas utilisant des fonctions à sens-unique ou de hachage, qui ne satisfont que des propriétés calculatoires.

Tout d'abord, nous proposons un nouveau schéma d'identification résistant aux attaques actives basé sur un problème combinatoire $\mathcal{NP}$-complet, le Problème des Perceptrons Permutés.

Puis, nous nous intéressons aux schémas de signature électronique et de signature en blanc. Pour cela, nous formulons un lemme générique, le «lemme de bifurcation». Ce lemme permet, dans un premier temps, de prouver que tout schéma de signature dérivé d'un protocole d'identification à divulgation nulle de connaissance face à un vérifieur honnête est existentiellement infalsifiable face aux attaques à messages choisis adaptatives. Une nouvelle version de ce «lemme de bifurcation» s'applique ensuite au concept plus complexe des signatures en blanc. Nous prouvons alors, pour un grand nombre de schémas de signature en blanc dérivés de protocoles d'identification à témoin indistinguable, l'impossibilité d'une falsification supplémentaire, même selon des attaques parallèles.

Enfin, nous appliquons ces derniers résultats sur les signatures en blanc à la monnaie électronique. Nous présentons alors un schéma de monnaie électronique «off-line» respectant l'anonymat avec des preuves formelles de sécurité aussi bien pour l'utilisateur que pour la Banque.

Abstract

The aim of cryptography is to protect communications, using encryption and authentication protocols. But what is the real security of proposed schemes against clever attackers? In this dissertation, we present provably secure cryptographic schemes in the random oracle model. This model, formalized in 1993 by Bellare and Rogaway, opens a way towards formal security proofs for schemes using one-way or hash functions, which satisfy only computational properties.

First, we present a new identification scheme, provably robust against active attacks, based on a combinatorial $\mathcal{NP}$-complete problem, the Permuted Perceptrons Problem.

Then, we focus on digital signatures and blind signatures. Here, we derive a generic lemma, the "forking lemma". It allows us to prove that any signature scheme derivated from a fair verifier zero-knowledge identification protocol is existentially unforgeable against adaptively chosen messages attacks. An improvement of this "forking lemma" allows its application for the more complex concept of blind signatures. It then provides the proof that a large number of blind signature schemes derivated from witness indistinguishable identification protocols cannot admit the so-called "one-more forgery", even under parallel attacks.

Finally, we apply the results about blind signatures to electronic cash. We present an "off-line" electronic payment system protecting privacy and we give formal proofs of its security for the bank and the users.

Remerciements

Je voudrais profiter de ce mémoire de thèse pour remercier tous ceux qui ont contribué, de près ou de loin, à son accomplissement. Je commencerai tout d'abord par Jacques Stern qui, après être passé par les stades d'examinateur d'informatique lors de mon oral au concours d'entrée à l'École Normale Supérieure, de professeur de logique en première année, puis de complexité en deuxième année, m'a accueilli dans son équipe pour diriger mes recherches, m'aider et me conseiller tout au long de ma thèse. Nos nombreuses conversations sont à l'origine de la plupart des résultats qui vont suivre. Je l'admire tout particulièrement pour son courage et sa persévérance face à certaines présentations de mes preuves «assez» obscures (n'est-ce pas Florent !). Ses lectures «commentées» de mes diverses rédactions m'ont toujours été très profitables et m'ont très souvent permis de les simplifier et de les clarifier. J'espère avoir acquis une partie de sa rigueur et de sa clarté qui en font un directeur de recherche exemplaire.

L'environnement du LIENS, et de toute l'École Normale Supérieure en général, fut indéniablement très positif tout au long de mes études. Je tiens donc à en remercier tous les élèves, secrétaires, enseignants et chercheurs. L'équipe du GRECC fut particulièrement sympathique. Je remercie chaleureusement toutes les personnes avec qui j'ai eu de nombreux travaux et discussions intéressants : Claude Crépeau, Ludovic Caudal, Philippe Béguin, Jean-Marc Couveignes, Jean-Bernard Fischer, Louis Granboulan, Philippe Hoogvorst et Thomas Pornin. Merci aussi à Florent Chabaud et Serge Vaudenay avec qui j'ai eu le plaisir de collaborer pour divers travaux, puis à Guillaume Poupard qui a accompli la tâche ardue de programmer le protocole d'identification PPP sur une carte à microprocesseur de capacité réduite. Ses résultats font d'ailleurs l'objet du paragraphe 4.8. Je le remercie aussi, ainsi que Phong Nguyen, pour leur relecture précieuse de cette thèse.

Je remercie aussi les différents visiteurs du GRECC, Antonella Cresti, Lars Knudsen, Holger Petersen, mais aussi Stefan Brands pour avoir consacré une partie de son temps, lors d'un passage à l'ENS, à expliquer son schéma de monnaie électronique en présentant ouvertement ses inconvénients. C'est à la suite de cet entretien que sont apparus les concepts de sécurité des signatures en blanc, puis les preuves.

J'en profite également pour remercier les partenaires du groupe, David M'Raïhi, pour nos intéressantes conversations (mais aussi pour son aide et son soutien lors de notre séjour en Corée ; sans lui, Asiacrypt '96 se serait sans doute transformé en cauchemar !), et David Naccache de l'entreprise GEMPLUS. Je suis également reconnaissant envers le CNET, et notamment Henri Gilbert, le CELAR, ainsi qu'envers la DRET pour m'avoir permis de publier un travail de recherche effectué à sa demande.

Je tiens à exprimer ma gratitude envers Brigitte Vallée pour sa gaieté et son soutien constants. Elle a suivi avec intérêt mes travaux tout au long de ma thèse. C'est elle qui m'a

fait découvrir le monde de la recherche et qui m'a fait l'apprécier. Le stage de DEA que j'ai effectué sous sa direction m'a permis de découvrir la cryptographie sous de nombreux angles : l'aspect recherche académique, et l'aspect appliqué puisque ce stage s'inscrivait dans le cadre d'un contrat avec le SEPT.

Ceci m'offre aussi l'occasion de remercier Marc Girault pour son aide non seulement au cours de ce stage, mais également tout au long de ma thèse et pour avoir accepté de siéger dans le jury.

Je remercie également tous les membres du GREYC, et sympathisants, avec qui j'ai eu d'intéressantes discussions au cours de mes passages à l'Université de Caen, et qui m'ont accueilli à plusieurs reprises à leurs séminaires et groupes de travail, notamment Claude Carlet, pour nos nombreuses conversations lors de nos retours sur Paris, Étienne Grandjean, Jean-Pierre Tillich et Philippe Toffin.

Je ne saurais oublier Robert Cori et Philippe Flajolet qui ont également accepté d'être membres du jury. Puis, je remercie Mihir Bellare, Guy Robin et Claus Schnorr pour avoir effectué le rude travail de rapporteur, leurs commentaires me furent fort utiles.

Je voudrais également remercier toutes les personnes qui m'ont soutenu tout au long de ces années, bien évidemment mes parents, mais également Nelly qui m'a supporté, des heures durant, rivé au clavier de mon ordinateur ou me débattant avec une des preuves qui vont suivre.

Enfin, je remercie tous ceux que j'aurais pu involontairement oublier.

Avant-Propos

Cette thèse a pour but de présenter des preuves de sécurité pour divers schémas d'authentification. Nous verrons tout d'abord le plus simple de ces schémas, l'identification. Puis nous traiterons les schémas de signature électronique et les schémas de signature en blanc. Enfin, nous appliquerons les techniques précédentes au cas plus complexe, et très sensible actuellement, de la monnaie électronique.

Ainsi, après une introduction générale à la cryptographie (chapitre 1), un chapitre préliminaire (chapitre 2) présentera les notions nécessaires de la théorie de la complexité, ainsi que le contexte dans lequel nous nous placerons par la suite. En effet, la grande majorité de nos résultats seront énoncés dans le modèle de l'oracle aléatoire formalisé par Mihir Bellare et Phillip Rogaway. Nous utiliserons donc des fonctions de «hachage» supposées parfaitement aléatoires. Nos résultats prouveront alors, tout d'abord la validité des «design» proposés, puis la sécurité des protocoles pratiques face aux attaques génériques, indépendantes des fonctions de hachage utilisées.

Les deux chapitres suivants (chapitres 3 et 4) traiteront du problème «simple» de l'identification. Ce problème est rencontré dès lors qu'un contrôle d'accès, ou d'identité, est nécessaire. Les propriétés de sécurité souhaitées seront définies et les schémas les plus célèbres seront présentés. Alors, nous détaillerons tout particulièrement le protocole PPP basé sur le problème combinatoire du même nom, PPP pour «Permuted Perceptrons Problem» ou «Problème des Perceptrons Permutés». Pour cela, nous commencerons par classifier ce problème au sens de la théorie de la complexité, puis nous en étudierons la difficulté pratique afin de définir un protocole d'identification «à divulgation nulle de connaissance», sûr et efficace. Nous verrons que ce protocole permet une identification complète à l'aide d'une carte à microprocesseur de capacité restreinte.

Le chapitre 5 concerne les schémas de signature électronique. Ils permettent, comme la signature manuscrite apposée au bas d'une lettre, de certifier l'identité de l'auteur d'un message ainsi que l'intégrité de ce message. Nous proposons ici une technique générique de preuve de sécurité pour une grande famille de tels schémas contre les falsifications existentielles selon des attaques adaptatives à messages choisis. Cette famille regroupe les schémas de signature dérivés de protocoles d'identification «à divulgation nulle de connaissance face à un vérifieur honnête» tels que les schémas de signature de Fiat-Shamir, de Schnorr ou une variante du schéma de El Gamal et toute signature à base de PKP, SD, CLE ou PPP.

Au cours du chapitre 6, nous présentons les premiers schémas de signature en blanc prouvés sûrs. De tels schémas permettent à un individu d'obtenir, d'une autre personne, sa signature d'un message sans lui révéler ni le message, ni la signature obtenue. Cette primitive est essentielle dans les protocoles nécessitant l'anonymat, tels que ceux de mon-

naie ou de vote électroniques. Nous prouvons notamment que les schémas de signature en blanc dérivés de protocoles d'identification «à témoin indistinguable» n'admettent pas de falsifications supplémentaires. Ceci permet alors de garantir à la Banque, lors d'une utilisation en monnaie électronique, qu'après un retrait de dix francs, l'utilisateur possède exactement dix francs, et pas plus. Jusqu'à présent, cette propriété était simplement supposée.

Enfin, le chapitre 7 propose une variante des protocoles de monnaie électronique présentés par Brands et Schoenmakers qui intègre les résultats précédents sur la signature en blanc. Nous pouvons alors exhiber certaines preuves de sécurité qui sont les premières fournies pour ce genre de schémas.

Table des matières

Table des figures

Signatures électroniques

Signatures en blanc

Monnaie électronique

Chapitre 1

Introduction à la cryptographie

1.1 Présentation générale

La cryptologie, et plus particulièrement la cryptographie, qui est l'art de cacher l'information, est une science très ancienne. Déjà au IX^{e} siècle avant notre ère, les Spartiates utilisaient des scytales pour assurer le secret de leur correspondance militaire. Dans toute l'Histoire, le combat entre la cryptographie et la cryptanalyse, dont le but est de retrouver le texte clair à partir d'un texte chiffré, a joué un grand rôle. Le livre historique de David Kahn [59] relate d'ailleurs un certain nombre d'anecdotes à ce sujet. L'exemple le plus célèbre est la spectaculaire cryptanalyse d'«Enigma», la machine allemande de chiffrement pendant la seconde guerre mondiale, par Alan Turing [54], l'un des pères de l'informatique. En effet, il a formalisé les ordinateurs en calculateurs universels connus sous le nom de machines de Turing [115, 44].

1.1.1 Définitions

Les définitions de tous les termes techniques seront regroupés dans le glossaire à la fin de ce mémoire, page 140. Nous allons, cependant, commencer par donner une définition succincte des principaux termes qui seront indispensables dès cette introduction.

Chiffrement : transformation appliquée à un message pour assurer sa *confidentialité*.

Clé : donnée qui personnalise un protocole cryptographique. Cette donnée peut être confidentielle, on parle dans ce cas de «clé secrète» ou privée. Elle peut également être connue de tous, on parle alors de «clé publique».

Confidentialité : garantie que seules les personnes autorisées ont accès à l'information. Pour cela, on utilise des procédés de *chiffrement*.

Contrôle d'accès/Identification : preuve d'identité.

Déchiffrement : transformation inverse du *chiffrement* qui consiste à retrouver l'information initiale contenue dans le message chiffré à l'aide d'une clé de «déchiffrement».

Décryptage : action de «casser» le *chiffrement* d'une information, et donc de retrouver le texte clair d'un chiffré, sans connaître la clé de «déchiffrement».

Fonction à sens-unique : fonction simple à calculer mais difficile à inverser.

Fonction à sens-unique à trappe : fonction à sens-unique. Mais une information supplémentaire appelée «trappe» permet une inversion aisée.

Intégrité : garantir l'intégrité d'un message signifie protéger ce message contre des modifications intentionnelles ou non, ou, tout du moins, permettre de détecter de telles transformations.

Machine de Turing : machine abstraite capable d'effectuer n'importe quel calcul. C'est une formalisation d'un «ordinateur». Une machine de Turing possède un ruban de travail, par exemple une bande magnétique, sur lequel est initialement écrite une donnée x, un ruban aléatoire qui fournit une séquence de bits aléatoires nécessaires à son fonctionnement si cette machine est probabiliste, puis une fonction de changement d'état qui décrit le «calcul» à effectuer. Une telle machine de Turing est dite polynomiale s'il existe un polynôme P tel que le temps d'exécution est borné par $P(|x|)$, où $|x|$ est la longueur, exprimée en bits, de x. Une définition plus formelle sera donnée dans le chapitre 2.

Signature : donnée de faible taille jointe à un message, qui prouve l'identité de l'auteur et qui garantit l'intégrité du message.

1.1.2 Présentation des acteurs

Après ces définitions, il convient de présenter les protagonistes de cette thèse, et de toute la cryptographie en général. Ces acteurs illustreront les protocoles que nous présenterons par la suite. Il ne faudra cependant pas oublier que ces acteurs à visages et noms «humains» seront, dans la vie réelle, des ordinateurs ou des cartes à microprocesseur, c'est-à-dire avec une puissance de calcul réelle, mais toujours limitée. Nous les formaliserons par des machines de Turing Polynomiales Probabilistes.

Alice et Bob seront deux interlocuteurs honnêtes. C'est-à-dire qu'ils utiliseront les protocoles suivant leur spécification.

Alice   Bob

Charlie, quant à lui, sera le «méchant». Il mettra en œuvre tous les moyens qu'il aura à sa disposition pour obtenir des informations secrètes, pour falsifier une identité, en bref, pour faire échec aux divers protocoles.

Charlie 

1.1.3 Un peu d'Histoire

Depuis l'invention de l'écriture, le besoin de sécurité est motivé par les problèmes de confidentialité et d'intégrité. Jusqu'à très récemment, deux principes gouvernaient le chiffrement – et deux seulement, quelles que soient les complications imaginées au cours des siècles par les spécialistes du chiffre :

- les permutations : modifier l'ordre des lettres du texte clair.
  Les scytales des Spartiates consistaient en des bâtons sur lesquels étaient enroulées des lanières de cuir. L'expéditeur écrivait son message sur les spires de la lanière. Celle-ci, déroulée, portait un texte inintelligible que le destinataire pouvait lire en l'enroulant de nouveau sur un bâton de même diamètre. Le diamètre était une sorte de «clé».

- les substitutions : remplacer les lettres, ou des blocs de lettres, du texte clair par d'autres caractères.
  Pendant la guerre des Gaules, Jules César usait d'un procédé de substitution qui garde son nom : l'«alphabet de César». Il consistait en un décalage constant et de faible amplitude des lettres de l'alphabet (dans ce cas précis, César utilisait un décalage de 3 lettres). L'amplitude du décalage aurait pu représenter une «clé secrète».

D'après la théorie de l'information définie par Shannon [105, 106], ces deux principes, les permutations, ou diffusion, et les substitutions, ou confusion, suffisent à gommer les redondances dans un texte.

Il est clair que dans les deux procédés précédemment décrits (les Scytales et l'Alphabet de César), la sécurité résidait, non pas dans une «clé secrète», mais dans le secret de l'algorithme lui-même. On parle alors d'algorithmes «restreints». En effet, pour l'Alphabet de César, le nombre de clés étant particulièrement réduit (26 au maximum), la recherche exhaustive aurait été parfaitement faisable à la main.

Pendant les deux guerres mondiales, le combat entre la cryptographie et la cryptanalyse fut décisif sur le plan militaire. En effet, pendant la deuxième guerre mondiale, les Anglais ont cassé le code allemand «Enigma» [59, 54]. Quant au code diplomatique japonais «PURPLE», il a été cassé par les États-Unis [59]. Ainsi, jusqu'à une époque récente, l'enjeu et les conséquences étaient tels que les recherches sont restées protégées par le secret militaire. La sécurité reposait principalement sur le secret des procédés employés, et sur la difficulté d'effectuer des essais répétitifs de manière systématique. L'apparition de l'ordinateur et des réseaux informatiques a bouleversé les concepts fondamentaux de la cryptologie :

- l'ordinateur offre une puissance de calcul aux cryptanalystes, leur permettant de progresser de manière significative. Un certain nombre de recherches exhaustives sont alors réalistes.

- les réseaux téléinformatiques ont rendu nécessaire la standardisation des moyens cryptographiques. L'utilisation d'algorithmes restreints devient alors illusoire.

- Les réseaux ont créé des problèmes nouveaux comme celui du *contrôle d'accès*.

Les besoins n'étaient plus uniquement diplomatiques ou militaires, ou des besoins personnels isolés, mais devenaient des besoins publics dans la vie quotidienne. Ainsi, désormais, la cryptologie est une discipline de recherche publique de l'informatique théorique.

La parution de l'article de Whitfield Diffie et Martin E. Hellman [31], en 1976, annonça les débuts d'une nouvelle ère en cryptographie, avec la notion de «clé publique» et de cryptographie asymétrique. En 1985, la théorie du «*zero-knowledge*» [48] ou preuves à divulgation nulle de connaissance, fut un nouveau grand pas. Ces deux notions seront plus amplement développées dans la suite.

1.2 Besoins et objectifs de la cryptographie

La cryptographie tente de satisfaire un grand nombre de besoins. Le tout premier était et reste la confidentialité. Ce problème est résolu par l'utilisation d'algorithmes de chiffrement et de déchiffrement. L'algorithme de chiffrement transforme le message pour le rendre incompréhensible de tous. Quant à l'algorithme de déchiffrement, il rétablit le message sous sa forme originale.

1.2.1 Chiffrement/déchiffrement

Depuis maintenant 20 ans, deux concepts s'affrontent. Chacun a ses avantages et ses inconvénients. Le chiffrement symétrique (ou conventionnel) consiste en un partage de données secrètes (ou clé secrète) entre les deux interlocuteurs, leur permettant de chiffrer et de déchiffrer. La notion de clé publique, présentée par Diffie et Hellman [31], en 1976, a entraîné l'apparition de protocoles dits «asymétriques» évitant ainsi ce partage de données secrètes.

1.2.1.1 Chiffrement symétrique

Dans la cryptographie conventionnelle, les clés de chiffrement et de déchiffrement sont identiques, donc partagées, mais gardées secrètes par l'expéditeur et le destinataire. À chaque couple d'utilisateurs est alors associée une clé k. Le procédé de chiffrement est dit *symétrique* (voir figure 1.1).

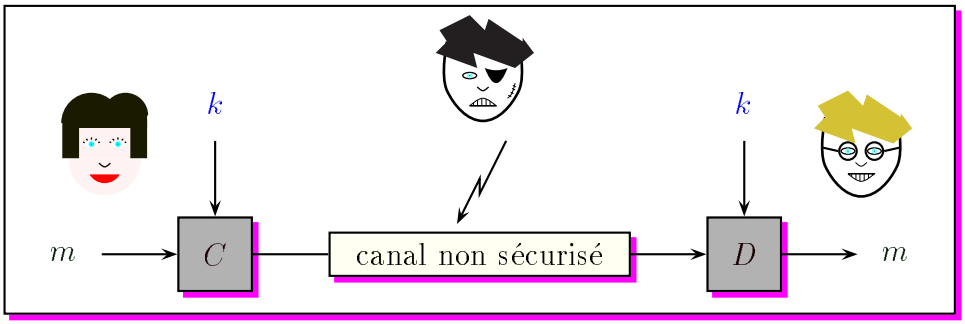

Fig. 1.1 – *Chiffrement symétrique*

Exemple 1 (Chiffre de Blaise Vigenère). Le chiffre de Vigenère repose sur un procédé de substitution polyalphabétique. Soient $M = (M_i) \in \{0, \ldots, 25\}^*$, le message à chiffrer, vu comme une suite de lettres, et $K = (k_0, \ldots, k_{m-1}) \in \{0, \ldots, 25\}^m$, la clé secrète. Le texte chiffré est alors $C = (C_i)$ où $C_i = M_i + k_{i \bmod m} \bmod 26$.

M = «LE CHIFFRE DE BLAISE VIGENERE» avec $K = (10, 7, 8, 4, 17, 20)$

M	L	E	C	H	I	F	F	R	E	D	E	B	L	A	I	S	E	V	I	G	E	N	E	R	E
M_i	11	4	2	7	8	5	5	17	4	3	4	1	11	0	8	18	4	21	8	6	4	13	4	17	4
K	10	7	8	4	17	20	10	7	8	4	17	20	10	7	8	4	17	20	10	7	8	4	17	20	10
C_i	21	11	10	11	25	25	15	24	12	7	21	21	21	7	16	22	21	15	18	13	12	17	21	11	14
C	V	L	K	L	Z	Z	P	Y	M	H	V	V	V	H	Q	W	V	P	S	N	M	R	V	L	O

C = «VL KLZZPYM HV VVHQWV PSNMRVLO»

Exemple 2 (Chiffre de Gilbert Vernam [119]). Le seul chiffre qui soit inconditionnellement sûr, c'est-à-dire dont la sécurité ne repose sur aucune hypothèse, est le chiffre de Vernam. Il utilise la technique du masque jetable (ou *one-time pad*). Pour chiffrer un message M de longueur n, Alice et Bob doivent partager une clé commune K, chaîne aléatoire de n bits. Le chiffré est alors $C = M \oplus K$, où $\oplus$ représente le «ou exclusif» bit à bit. Bob peut retrouver le message initial à partir du message chiffré C avec $M = C \oplus K$.

Cette technique est inconditionnellement sûre sous réserve que la chaîne commune K soit parfaitement aléatoire. L'inconvénient majeur de cette technique est qu'il faut partager des chaînes aléatoires extrêmement longues. Ceci n'est envisageable que pour des communications à faible débit, ultra-secrètes. Le «téléphone rouge» entre les États-Unis et l'ex-Union Soviétique utilisait (?) ce procédé.

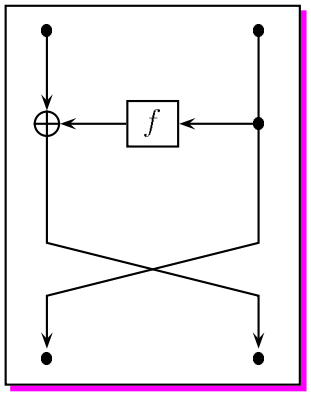

Fig. 1.2 – *Schéma de Feistel*

Exemple 3 (Data Encryption Standard). Un exemple plus récent est le **DES** [116], système de chiffrement développé par IBM en 1977 sur commande du gouvernement américain. Ce système est basé sur le schéma de Feistel (voir figure 1.2), utilisé initialement

dans la fonction Lucifer [111] proposée par Horst Feistel. Il consiste en un réseau de permutations-substitutions, comme défini par Shannon [106], avec des boîtes de confusion, plus connues sous le nom de «S-box». Encore 20 ans après, le DES résiste toujours aux cryptanalystes. Cependant, la faible taille de la clé secrète, 56 bits, risque d'entraîner une «mort» prématurée. En effet, la recherche exhaustive, qui est quasiment la meilleure attaque, devient de plus en plus réaliste.

L'inconvénient majeur du chiffrement symétrique est la nécessité de distribuer des clés secrètes. Il est bien clair que partager ainsi le secret le rend beaucoup plus vulnérable.

1.2.1.2 Chiffrement asymétrique

Dans la cryptographie à clé publique, la clé de chiffrement, K_p, est publique et la clé de déchiffrement, K_s, secrète. Le procédé est dit *asymétrique* (voir figure 1.3).

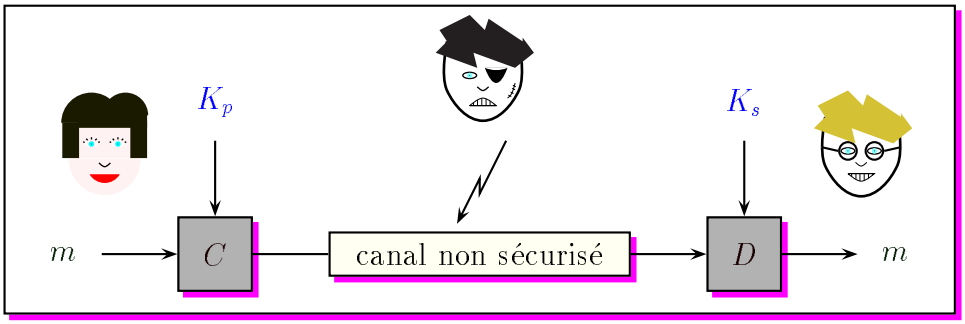

Fig. 1.3 – *Chiffrement asymétrique*

Ce sont Whitfield Diffie et Martin E. Hellman [31] qui, en 1976, ont émis l'idée d'une cryptographie à «clé publique». Ronald Rivest, Adi Shamir et Leonard Adleman ont apporté une solution en 1978, connue sous le nom de RSA [97], largement utilisée désormais. Cette méthode repose sur un problème de théorie des nombres, supposé difficile :

Hypothèse RSA. *Pour tout triplet (N, k, x), où $N, k \in \mathbb{N}$ et $x \in (\mathbb{Z}/N\mathbb{Z})^\star$, le calcul d'une racine k-ième de x modulo N est difficile sans la connaissance des facteurs premiers de N.*

Description du chiffrement RSA

– Pour initialiser le protocole, Alice, comme tous les autres utilisateurs, calcule le produit de deux grands nombres premiers de tailles semblables, $N = p \cdot q$. Cet entier N est appelé *module RSA*. Elle choisit ensuite son exposant public e, appelé «exposant de chiffrement», et son exposant secret s vérifiant $e \cdot s = 1 \bmod \varphi(N)$ où φ est la fonction indicatrice d'Euler, $\varphi(N) = (p-1)(q-1)$.

Remarque. D'après le théorème d'Euler, pour tout nombre $x \in (\mathbb{Z}/N\mathbb{Z})^\star$,

$$(x^e)^s = (x^s)^e = x \bmod N.$$

Elle publie alors (N, e) comme clé publique et conserve privée sa clé secrète s.

- Pour envoyer un message m à Alice, Bob utilise la clé publique (N, e) d'Alice pour calculer le message chiffré $C = m^e \mod N$. Nul ne peut prendre connaissance du message m, à moins d'être capable de calculer la racine e-ième modulo N.
- Seule Alice, grâce à sa clé secrète s, retrouve sans difficulté $m = C^s \mod N$.

Autres problèmes difficiles. L'application RSA(N,e) qui associe, à tout élément $x \in (\mathbb{Z}/N\mathbb{Z})^\star$, l'élément $x^e \mod N$ est une fonction à sens-unique. En effet, sans la connaissance des facteurs de N, l'inversion est calculatoirement impossible d'après l'hypothèse RSA. En revanche, la connaissance des facteurs de N, ou de s, permet une inversion aisée. Les facteurs de N, ou l'exposant s, sont appelés «trappe». On peut remarquer que l'application RSA(N,e) est plus qu'une «fonction à sens-unique à trappe», car c'est de plus une permutation de $(\mathbb{Z}/N\mathbb{Z})^\star$. Cette propriété en fait la famille de fonctions la plus utilisée en cryptographie.

En fait, tout problème «difficile» à trappe peut être utilisé. C'est-à-dire, tout problème difficile à résoudre (impossible calculatoirement) à moins de connaître une information supplémentaire. D'autres problèmes, basés sur la théorie des codes correcteurs d'erreurs, ont été proposés, notamment par McEliece [65].

Taille des problèmes. La taille du problème utilisé est déterminée en fonction de la confidentialité souhaitée ainsi que de la durée de vie des données chiffrées et, bien entendu, de la difficulté du problème. On peut réclamer un temps d'attaque allant de quelques années sur une machine personnelle jusqu'à l'âge de l'Univers sur le parc mondial des machines. Il faut, de plus, tenir compte de l'évolution fulgurante de la puissance des machines. La notion de «problème difficile» sera précisée un peu plus loin.

Les avantages et les inconvénients. L'avantage majeur de la cryptographie à clés publiques est la gestion des clés. Il n'y a plus de données secrètes à partager et beaucoup moins de clés sont nécessaires. En effet, pour la cryptographie conventionnelle, chaque couple d'individus doit partager une clé secrète (de l'ordre de $n^2/2$ clés secrètes, chacune partagée entre 2 individus). Dans le cas de la cryptographie à clés publiques, chaque individu possède son couple de clés publique et secrète (seulement n couples de clés). Sa clé secrète n'a pas à être partagée, le secret est donc plus aisément gardé. L'ajout d'un utilisateur s'en trouve simplifié.

Cependant, le chiffrement symétrique a l'avantage d'être, en général, beaucoup plus rapide. Tous les inconvénients peuvent être supprimés par la combinaison des deux systèmes. Lorsqu'Alice veut établir une communication avec Bob, ils se mettent d'accord sur une clé, dite de «session», à l'aide d'un procédé asymétrique, puis utilisent cette clé de session secrète et commune avec un procédé conventionnel plus rapide pour dialoguer.

1.2.2 Échange de clés

Cette mise en accord de clé de session peut bien évidemment être effectuée à l'aide d'un procédé de chiffrement asymétrique, mais on peut également utiliser un protocole

spécifique à cet usage, appelé protocole d'*échange de clés*.

La première solution fut proposée par Whitfield Diffie et Martin E. Hellman [31] en 1976, ce fut d'ailleurs le premier protocole cryptographique à clé publique. Il est présenté figure 1.4. À l'aide de ce protocole, Alice et Bob se sont mis d'accord sur une clé de

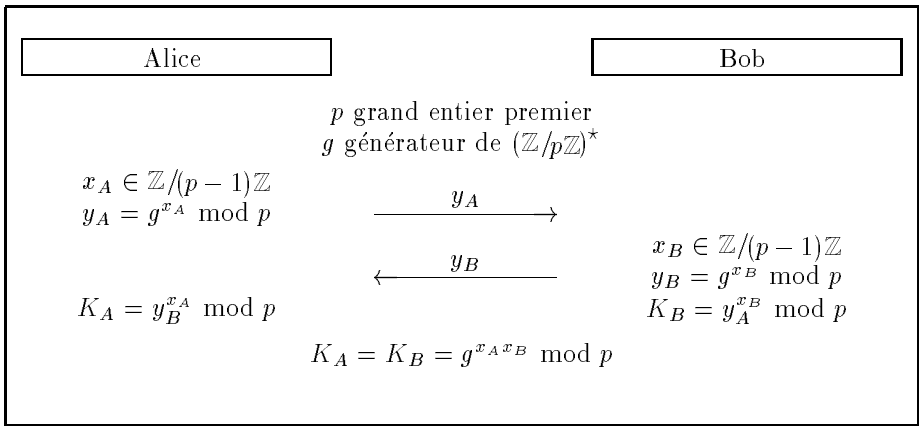

Fig. 1.4 – *Protocole d'échange de clé de Diffie–Hellman*

communication en ne se transmettant aucune donnée confidentielle. Cependant, il ne faut pas d'intervention active de la part de Charlie qui modifierait les informations transmises par le canal. En effet, une attaque classique «par le milieu» (ou «de la mafia») permettrait à Charlie d'établir une clé secrète de communication avec Bob, tout en lui laissant croire à ce dernier qu'il la partage avec Alice. De façon symétrique, il peut établir cette clé avec Alice. Et alors, Charlie pourrait intercepter et déchiffrer les messages venant d'Alice ou de Bob. Un contrôle d'identité se trouve être indispensable en télécommunications.

1.2.3 Intégrité

Le problème de l'intégrité est également un problème ancien mais qui s'accentue avec les documents informatiques. En effet, la copie, la modification puis la diffusion du document modifié sont choses faciles. Il est donc important de garantir qu'un document n'a pas été modifié. Ce problème peut apparaître dans différentes situations :

- partage de ressources (par exemple sur un disque). On veut être sûr que personne n'a modifié, volontairement ou non, les données.

- envoi de messages sur un canal non sécurisé. On veut être sûr que le message reçu est bien le message envoyé.

Pour cela, on utilise des fonctions de hachage qui réduisent une liste de données de longueur quelconque en un bloc de longueur prédéfinie appelé «condensé» ou «empreinte». Cette empreinte peut alors être stockée sur un autre support très réduit (disquette) ou

envoyée par un autre canal. Les fonctions de hachage à usage cryptographique doivent satisfaire les deux propriétés suivantes :

1. sens unique (*one-wayness*) : il est impossible de trouver un antécédent en un temps raisonnable. Ceci empêche un intrus de modifier intentionnellement le contenu d'un fichier ou d'un message en conservant la même empreinte.

2. résistance aux collisions (*collision-freeness*) : il est impossible de trouver, en un temps raisonnable, deux éléments ayant la même image. Ceci empêche l'auteur, lui-même, de créer deux fichiers ou deux messages différents ayant le même condensé. Il pourrait ensuite nier le premier en exhibant le deuxième. Cette notion est nécessaire en signature, lorsque l'on se contente de signer l'empreinte.

Les exemples les plus célèbres sont dans la famille des «Message Digest» MDx, avec MD2 [60], MD4 [94, 95], MD5 [96] et SHA (pour *Secure Hash Algorithm*), la fonction de hachage du standard SHS (pour *Secure Hash Standard*) [70, 71, 73], définie pour une utilisation conjointe au DSA (pour *Digital Signature Algorithm*) [69, 72]. Cependant, après de nombreuses attaques partielles [29], MD4 a récemment été cassé par Dobbertin [33]. C'est-à-dire que des collisions ont été trouvées pour la fonction MD4. La fonction de hachage MD5 commence également à être fortement menacée, suite à la découverte de collisions de la fonction de compression [32, 34].

1.2.4 Identification/Contrôle d'accès

Les liaisons informatiques permettent à tout individu d'accéder à des machines du monde entier. Ceci pose un problème de sécurité. Il faut contrôler l'identité de tout individu qui tente de se connecter. Pour cela, on utilise des protocoles d'identification qui consistent en une communication entre un prouveur (client) et un vérifieur (serveur). Le prouveur tente de convaincre le vérifieur de son identité. Cette communication peut être plus ou moins complexe selon les attaques que l'on veut contrer. Cette notion d'identification sera tout particulièrement développée dans le chapitre 3. Des schémas prouvés «sûrs» seront proposés aux chapitres 3 et 4.

1.2.5 Signatures

Un schéma d'authentification un peu plus général que l'identification est également devenu nécessaire. Il s'agit de la *signature électronique*. C'est un bloc de faible taille qui est ajouté à la fin d'un message pour convaincre le destinataire de l'identité de l'expéditeur ainsi que de l'intégrité du message reçu. La signature électronique doit, comme la signature manuscrite apposée au bas d'une lettre, convaincre le destinataire, mais également toute tierce personne. En fait, c'est le résultat d'un calcul effectué sur le message par l'expéditeur. Et ce calcul, lui seul peut l'opérer mais tout le monde peut le vérifier comme l'illustre la figure 1.5.

On peut déjà remarquer que toute permutation à sens-unique à trappe, telle RSA [97], permet la signature électronique. En effet, si C et D sont respectivement les algorithmes de chiffrement et de déchiffrement opérant sur un même ensemble $\mathcal{M}$, alors pour tout

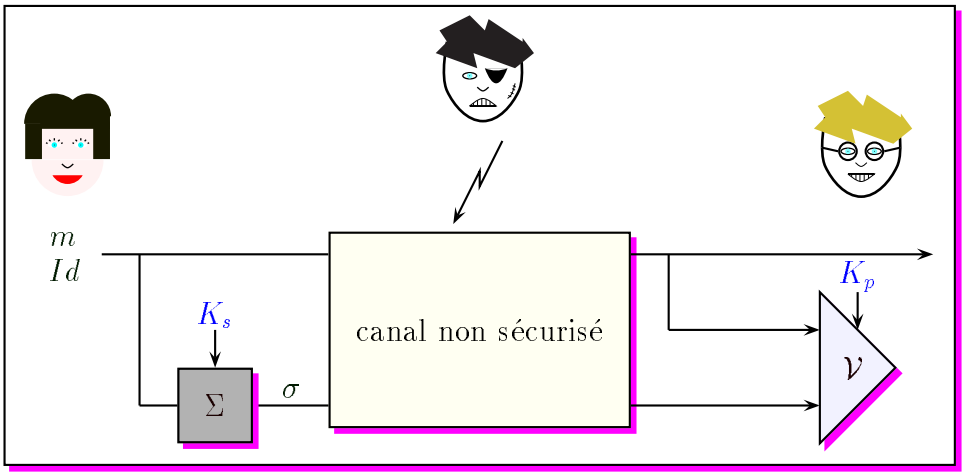

Fig. 1.5 – *Schéma de signature*

message $m \in \mathcal{M}$, $D_{K_s}(E_{K_p}(m)) = E_{K_p}(D_{K_s}(m)) = m$. Ainsi, en «déchiffrant» le message m, avec sa clé secrète K_s, Alice obtient un bloc Sig, qu'elle seule était capable de calculer, tel que $E_{K_p}(Sig) = m$. Et ce dernier calcul, tout le monde peut le vérifier puisque K_p est la clé publique.

En général, pour obtenir une signature de petite taille, on applique ce procédé, non pas au message m, mais à son condensé par une fonction de hachage H. Il est alors nécessaire que cette fonction H résiste aux collisions, sinon l'expéditeur peut envoyer un message m signé, tout en connaissant un message m', signifiant autre chose que m, fournissant le même condensé par H. S'il souhaite ensuite renier sa signature sur m, il peut déclarer qu'il n'a jamais signé m, mais m'.

Description de la signature RSA. Le schéma RSA [97] permet, à l'aide de la remarque ci-dessus, d'effectuer des signatures :

- Alice possède une clé publique (N, e) et une clé secrète s, appelée «clé de signature». La fonction de hachage, H, est commune à tout utilisateur.

- Pour signer un message m, Alice calcule $Sig = H(m)^s \bmod N$. Elle envoie alors le couple (m, Sig).

- Bob, mais également toute autre personne, peut vérifier, à l'aide de la clé publique (N, e) d'Alice, que $Sig^e = H(m) \bmod N$.

Les familles de permutations à sens-unique à trappe sont peu nombreuses, de plus, un schéma de signature électronique les utilisant est susceptible d'être détourné en protocole de chiffrement, ce qui n'est pas souhaitable pour des raisons de législation, on préfère donc utiliser des schémas de signature reposant simplement sur des problèmes «difficiles». Les plus célèbres utilisent le problème du logarithme discret, tels les schémas d'El Gamal [35], de Schnorr [98, 99] ou l'algorithme DSA (pour *Digital Signature Algorithm*) publié par le

NIST en août 1991, et adopté dans le standard DSS (pour *Digital Signature Standard*) [69, 72] en décembre 1994.

Description de l'algorithme DSA. Une signature, à l'aide de DSA s'effectue de la façon suivante :

- De façon universelle, sont publiés un grand premier p tel que $p-1$ possède un grand facteur premier q de 160 bits, et un élément g de $\mathbb{Z}/p\mathbb{Z}$ d'ordre q. La fonction H est une fonction de hachage sans collisions, publique. Dans la définition du standard [69, 72], il est spécifié l'utilisation de SHA.

- Alice possède une clé secrète $x \in \mathbb{Z}/q\mathbb{Z}$ et publie sa clé publique $y = g^x \bmod p$.

- Pour signer un message m, Alice choisit un entier aléatoire $k \in (\mathbb{Z}/q\mathbb{Z})^\star$ puis calcule $r = (g^k \bmod p) \bmod q$ ainsi que $s = k^{-1}(H(m) + xr) \bmod q$. Si $r = 0$ ou $s = 0$, ce qui est très peu probable, Alice choisit un nouvel élément k. Sinon, elle envoie le couple $(m, (r, s))$, où (r, s) est la signature.

- Bob, mais également toute autre personne, peut vérifier, à l'aide de la clé publique y d'Alice, que $0 < r < q$, $0 < s < q$ et si l'on pose $w = s^{-1} \bmod q$, $u_1 = wH(m) \bmod q$ et $u_2 = wr \bmod q$, alors $r = (g^{u_1} y^{u_2} \bmod p) \bmod q$.

Cette deuxième famille de schémas de signature basés sur des problèmes «difficiles» sera étudiée en détail dans le chapitre 5. Une technique générale de preuve y sera présentée. Les exemples les plus classiques seront analysés.

1.2.6 Schémas de signatures dérivés

Plusieurs variantes des schémas de signatures existent :

signatures à vérifieur désigné (designated confirmer signatures [23]) : l'auteur de la signature peut désigner les personnes à convaincre de la validité de la signature. Cette conviction acquise ne pourra être transmise à personne.

signatures indéniables (undeniable signatures [26, 22]) : la signature d'un message ne peut être vérifiée qu'avec la collaboration du signeur. Une signature valide ne pourra être reniée. Mais la conviction acquise par le vérifieur ne pourra être transmise. En revanche, une signature invalide pourra être prouvée telle par le signeur.

signatures indéniables convertibles (convertible undeniable signatures [7]) : c'est un schéma de signature indéniable dans lequel la révélation d'une information transforme la signature indéniable en signature électronique que tout le monde peut vérifier.

signatures en blanc (blind signatures [19]) : Alice accepte d'aider Bob à signer un message en son nom. Malgré cette aide, Alice n'aura aucune information sur le message signé, ni sur la signature obtenue. Cette notion, très utilisée dans les applications nécessitant l'anonymat, telles que la monnaie ou les élections électroniques, sera étudiée dans le chapitre 6.

signatures en blanc à trappe (fair blind signatures [17]) : c'est un schéma de signature en blanc dans lequel un juge possède une information supplémentaire (trappe) lui permettant de relier les signatures obtenues aux interactions avec le signeur. Cette notion ouvre la possibilité de monnaie électronique où l'anonymat peut être levé par une autorité.

signatures sans échec (fail-stop signatures [117]) : une falsification est, bien entendu, *a priori* impossible, mais en cas de falsification de signature, Alice est capable de prouver que ce n'est pas elle qui a établi la signature.

1.2.7 Monnaie électronique

Dès 1982, David Chaum [19] a voulu fabriquer l'équivalent de l'argent liquide sous forme électronique. Il souhaitait conserver les mêmes propriétés, notamment l'anonymat. Sont alors apparues les «pièces de monnaie» électroniques et les signatures en blanc. Ces signatures en blanc sont toujours l'outil essentiel en monnaie électronique pour assurer l'anonymat. Elles seront étudiées au chapitre 6. On appliquera alors les résultats obtenus à la monnaie électronique dans le chapitre 7 pour proposer le premier schéma anonyme prouvé sûr.

Chapitre 2

Préliminaires

2.1 Les modèles de sécurité

Dans les chapitres suivants, nous allons analyser un certain nombre de schémas, et fournir des preuves de sécurité. Les preuves de sécurité seront relatives à la calculabilité ou non de certaines fonctions. Pour cela, nous allons supposer que certains problèmes ne sont pas solubles en pratique, c'est-à-dire qu'aucun calculateur n'est capable de les résoudre. Ensuite, nous allons réduire un tel problème au cassage d'un schéma cryptographique. Ainsi, une machine capable de casser le schéma cryptographique pourra être utilisée pour résoudre efficacement le problème dit insoluble, ce qui mènera à une contradiction.

Cependant, il ne faut pas que la réduction d'un problème à l'autre soit trop «coûteuse» en temps de calcul. De plus, il nous faut formaliser un calculateur afin d'avoir une évaluation précise du «temps de calcul». Nous allons donc utiliser des Machines de Turing, qui sont désormais le *mètre-étalon* de la complexité. Les définitions suivantes proviennent de la traduction [44] de l'article initial de Turing [115].

Définition 1 (Machine de Turing). Une machine de Turing consiste en :

- un *ruban*, avec une extrémité à gauche, infini à droite, divisé en cases de même taille. On peut penser, par exemple, à une bande magnétique suffisamment longue ;
- un ensemble fini de *symboles*, par exemple $0, 1, s, d, f$ (0 et 1 pour la numérotation binaire, s pour séparer deux expressions, d pour le début du ruban et f pour signifier que ce qui est à droite n'a plus d'importance). Ils seront inscrits et lus sur le ruban à raison d'un par case ;
- une *tête de lecture/écriture* qui se déplace sur le ruban. À chaque impulsion, la tête peut soit rester sur place, soit rétrograder (si possible) ou avancer, dans les deux cas d'une case. La tête a le droit de lire ou d'écrire dans la case qu'elle est en train de sonder. Pour rester concret, on peut penser que c'est la bande et non la tête qui bouge ;
- un ensemble fini d'*états*. Ces états permettent de distinguer plusieurs comportements possibles. Notamment l'état D représente le départ et l'état F symbolise la fin du «programme» et donc la lecture du résultat par un observateur extérieur ;

– un ensemble fini d'*instructions* : à chaque impulsion, en fonction du symbole c lu par la tête, en fonction de l'état courant S, la tête écrit un nouveau symbole c', effectue un déplacement noté -1 (gauche), $+1$ (droite) ou 0 (immobilité) et passe à l'état S'.

Le programme fonctionne ainsi : les données de départ sont inscrites sur le ruban, puis la machine est initialisée dans l'état D sur la première case. La machine peut alors

– se bloquer, si une situation imprévue se présente ;

– fonctionner pour toujours, si l'état F n'est jamais atteint ;

– s'arrêter dans l'état F. On peut alors lire le résultat.

Dans la suite, nous ne considérerons que des machines qui finissent normalement dans l'état F au bout d'un temps fini.

Définition 2 (Complexité). La complexité d'une machine de Turing est le nombre d'impulsions nécessaires avant d'atteindre l'état final F.

Définition 3 (Machine de Turing Polynomiale et Probabiliste). Une machine de Turing est dite Polynomiale si sa complexité est polynomiale en la taille des données de départ. C'est-à-dire qu'il existe un polynôme P tel que si «dxf» est l'état initial du ruban, le nombre d'impulsions avant d'atteindre l'état F est borné par $P(|x|)$.

Cette machine est de plus dite Probabiliste si, au cours de son exécution, elle a accès en lecture à un ruban supplémentaire, appelé «ruban aléatoire», contenant une liste de bits aléatoires. Il sert de source d'aléa.

Nous allons nous attacher à trouver des bornes inférieures sur la complexité d'un attaquant. Ces bornes seront asymptotiques et auront donc un intérêt au sens de la théorie de la complexité. Elles permettront alors essentiellement de valider le «design». Par exemple, dans le chapitre sur les schémas de signatures électroniques (voir chapitre 5), nous prouverons la sécurité d'une famille de schémas à laquelle le standard DSS n'appartient pas à cause de ce que nous pourrions appeler un «défaut» de fabrication. Et ce «défaut» a été utilisé par Vaudenay [118] pour présenter une fraude possible.

Cependant, nous tenterons d'optimiser les réductions afin d'obtenir des bornes les plus proches possible de la notion de sécurité «exacte» définie par Bellare et Rogaway [4]. Cela nous fournira alors une évaluation plus précise de la sécurité.

Bon nombre de schémas cryptographiques reposent sur des résultats de la théorie de la complexité. Par exemple, qu'est-ce qu'un problème «difficile» ? Cette notion existe de façon plus formelle dans la théorie de la complexité sous le nom de problèmes $\mathcal{NP}$-complets.

2.1.1 Les classes de complexité

2.1.1.1 Les classes $\mathcal{P}$ et $\mathcal{NP}$

Dans la théorie de la complexité, tous les problèmes sont classés dans différentes «classes de complexité» (voir figure 2.1). La première classe est la classe $\mathcal{P}$ qui regroupe les

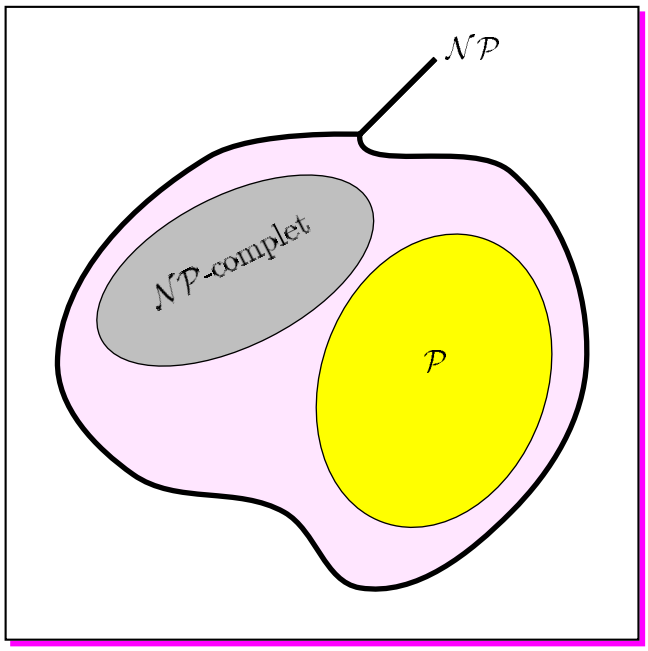

Fig. 2.1 – *Classes de complexité*

problèmes de décision qu'une **machine de Turing Polynomiale** peut résoudre. C'est-à-dire l'ensemble des langages dont les mots peuvent être reconnus en temps polynomial par une machine de Turing. Un exemple classique est le problème 2-SAT :

Problème 4 (La satisfaisabilité d'un ensemble de 2-clauses – 2-SAT).
Donnée : *Soit un ensemble $\mathcal{C}$ de 2-clauses (disjonctions de 2 littéraux) sur l'ensemble de variables U.*
Question : *Existe-t-il une distribution de vérité pour les variables de U telle que toutes les clauses de $\mathcal{C}$ soient satisfaites?*

Malheureusement, peu de problèmes admettent ainsi un «algorithme» de résolution polynomial en la taille du problème. Notamment si l'on passe aux 3-clauses, aucun algorithme polynomial n'est connu :

Problème 5 (La satisfaisabilité d'un ensemble de 3-clauses – 3-SAT).
Donnée : *Soit un ensemble $\mathcal{C}$ de 3-clauses (disjonctions de 3 littéraux) sur l'ensemble de variables U.*
Question : *Existe-t-il une distribution de vérité pour les variables de U telle que toutes les clauses de $\mathcal{C}$ soient satisfaites?*

Ce problème est l'exemple classique d'une classe de complexité un peu plus générale appelée la classe $\mathcal{NP}$. Cette classe regroupe les problèmes de décision que peut résoudre une **machine de Turing Polynomiale non Déterministe**. Une telle machine est encore plus hypothétique qu'une machine de Turing : dans un état S et lisant le symbole c, elle a le choix entre plusieurs instructions (toujours en nombre fini). À partir d'une configuration

initiale fixée, une machine *non déterministe* a plusieurs *exécutions* possibles. Une machine de Turing non déterministe *reconnaît* une propriété *Prop* si et seulement si pour toute entrée x elle finit par répondre «oui» pour une certaine exécution du programme et cela exactement quant $Prop(x)$ est vrai. Plus concrètement, on peut penser que cette machine a énormément d'intuition et devine l'exécution qui mène au résultat. En d'autres termes, $\mathcal{NP}$ est l'ensemble des problèmes pour lesquels la validité d'une solution se vérifie en temps polynomial.

Les problèmes du voyageur de commerce et du sac à dos sont d'autres exemples bien connus appartenant à la classe de complexité $\mathcal{NP}$. En voici deux autres qui auront leur importance dans la suite :

Problème 6 (Le problème des noyaux permutés – PKP).
Donnée : *Soient un entier p, une matrice A de taille $m \times n$ sur $\mathbb{Z}/p\mathbb{Z}$, puis un vecteur V de taille n sur $\mathbb{Z}/p\mathbb{Z}$.*
Question : *Existe-t-il un permuté V_π de V qui appartienne au noyau de A, c'est-à-dire tel que $A \cdot V_\pi = 0 \ mod\ p$?*
À la vue d'une permutation π, on peut aisément déterminer (en temps polynomial) si c'est effectivement une solution. C'est-à-dire si $A \cdot V_\pi = 0 \ mod\ n$.

Ce dernier exemple fera l'objet d'une étude toute particulière dans le chapitre 4.

Problème 7 (Le Problème des Perceptrons – PP).
Donnée : *Soit une matrice A de taille $m \times n$ à coefficients dans $\{-1, +1\}$.*
Question : *Existe-t-il un vecteur Y de taille n à coefficients dans $\{-1, +1\}$ tel que le vecteur produit $A \cdot Y$ ait toutes ses composantes positives?*

Remarque. Bien évidemment, 2-SAT appartient également à $\mathcal{NP}$. En revanche, il n'a pas été prouvé que 3-SAT, PKP ni **PP** appartiennent à $\mathcal{P}$.

Usuellement, on admet que ces deux classes sont distinctes. C'est cependant une grande question qui motive la recherche en théorie de la complexité :

$$\mathcal{P} \stackrel{?}{=} \mathcal{NP}.$$

Définition 8 ($\mathcal{NP}$-complet). Un problème $\mathcal{NP}$-complet est un problème X auquel tout problème de la classe $\mathcal{NP}$ est polynomialement réductible. C'est-à-dire que pour tout problème de la classe $\mathcal{NP}$, il existe une réduction qui transforme ce problème en le problème X en temps polynomial en la taille du problème.

Remarque. Si pour un problème $\mathcal{NP}$-complet X, on découvre une machine de Turing Polynomiale qui le résout, la classe $\mathcal{NP}$ se réduit alors à $\mathcal{P}$. D'où l'importance des problèmes $\mathcal{NP}$-complets. Soit l'ensemble des problèmes $\mathcal{NP}$-complets et $\mathcal{P}$ sont disjoints, soit ils sont égaux, et alors $\mathcal{P} = \mathcal{NP}$.

Exemple 4. PKP et 3-SAT sont des problèmes $\mathcal{NP}$-complets [43]. Il en est de même pour **PP** (voir chapitre 4).

Une bonne partie de la cryptographie repose sur la différence entre $\mathcal{P}$ et $\mathcal{NP}$. L'égalité remettrait en cause un certain nombre de résultats. En effet, cette différence nous permet d'affirmer qu'aucune machine de Turing polynomiale ne peut résoudre un problème $\mathcal{NP}$-complet. Par conséquent la résolution d'un problème $\mathcal{NP}$-complet, dans le cas le pire, nécessite un temps super-polynomial T, c'est-à-dire que T est plus grand que tout polynôme en n, la taille du problème. D'ailleurs, jusqu'à ce jour seuls des algorithmes exponentiels existent. C'est-à-dire que le temps T nécessaire à l'algorithme pour résoudre un problème $\mathcal{NP}$-complet dans le cas le pire est exponentiel en n. Si nous arrivons à faire reposer un protocole cryptographique sur les instances difficiles de ces problèmes $\mathcal{NP}$-complets alors la sécurité croît «exponentiellement» en la taille des clés. Ainsi, si une dimension est menacée, en l'augmentant légèrement, la résolution est vite mise hors de portée des super-calculateurs (voir figure 2.2).

Complexité	$n = 30$	Temps à 100 MHz	$n = 90$	Temps à 100 MHz
n	30	0.3 μs	90	0.9 μs
n^3	27 10^3	270 ms	729 10^3	7.3 ms
n^6	7 10^8	7.3 s	5 10^{11}	1.5 min
$e^{2n^{1/3}(\log n)^{2/3}}$	1.2 10^6	13 μs	41 10^9	6.8 min
2^n	10^9	10 s	10^{27}	4 10^{11} années âge de l'Univers

Fig. 2.2 – *Ordre de grandeur de complexité*

Les problèmes $\mathcal{NP}$-complets sont les problèmes précédemment appelés «difficiles». La communauté scientifique est intimement convaincue de la différence entre $\mathcal{P}$ et $\mathcal{NP}$, ainsi cherche-t-on à utiliser des problèmes $\mathcal{NP}$-complets pour la cryptographie plutôt que des problèmes de théorie des nombres, tels que la factorisation ou le logarithme discret, qui sont simplement dans $\mathcal{NP}$ et dont la complexité des meilleures attaques, pour le moment, est de l'ordre de $\exp(2n^{1/3}(\log n)^{2/3})$ (voir figure 2.2).

2.1.1.2 Le passage de $\mathcal{NP}$ à $\mathcal{IP}$

Nous allons formaliser un peu ces définitions intuitives. En fait, $\mathcal{NP}$ est l'ensemble des langages $\mathcal{L}$ pour lesquels il existe une machine de Turing polynomiale M à 2 arguments telle que $x \in \mathcal{L}$ si et seulement s'il existe un témoin w, de longueur polynomiale en la taille de x, tel que $M(x, w) = 1$. Le témoin w décrit l'exécution qui déterminise la Machine de Turing non Déterministe.

Définition 9. $\mathcal{L} \in \mathcal{NP}$ signifie qu'il existe une machine de Turing Polynomiale M

- *consistante* : si $x \in \mathcal{L}$, il existe un témoin w tel que $M(x, w) = 1$.
- *significative* : si $x \notin \mathcal{L}$, alors il n'existe aucun témoin w tel que $M(x, w) = 1$.

On peut voir également cette classe $\mathcal{NP}$ comme l'ensemble des langages dont l'appartenance ou la non appartenance d'un mot est prouvée interactivement : un prouveur P, ayant une puissance de calcul infinie, et un vérifieur V, qui répond en temps polynomial, ont un mot x sur leur ruban, P tente de prouver à V que $x \in \mathcal{L}$. Il faut noter que tous deux sont probabilistes.

Nous avons alors à faire à un système interactif de preuve [49], noté (P, V), entre deux machines de Turing (voir figure 2.3).

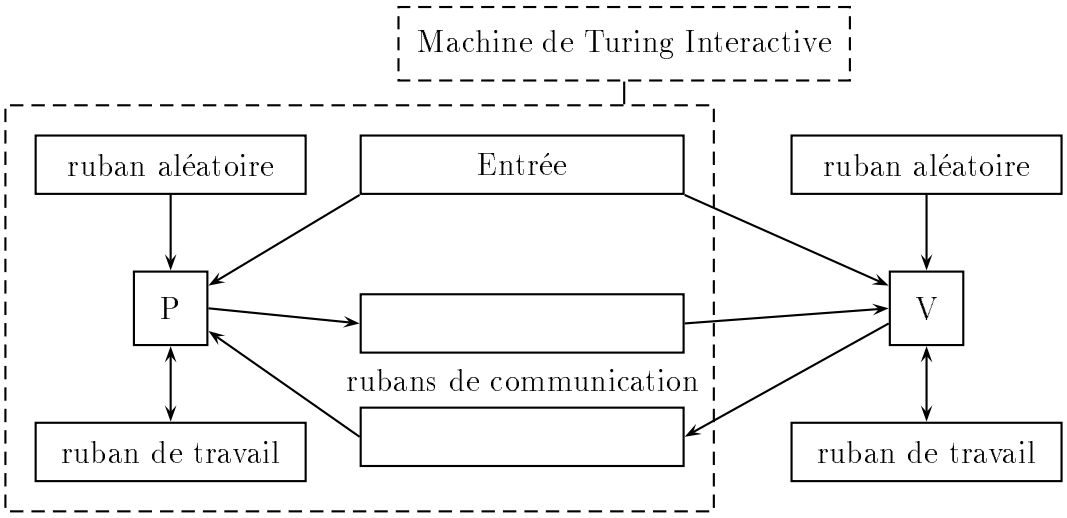

Fig. 2.3 – *Système interactif*

Notation 10. Nous noterons $Output(P, V)[x]$ la réponse de V à la fin du protocole interactif (P, V) sur l'entrée x. Cette réponse sera égale à 1 lorsque le vérifieur «accepte» la preuve, donc est convaincu du résultat. Dans le cas contraire, cette réponse est égale à 0.

Définition 11. $\mathcal{L} \in \mathcal{NP}$ signifie qu'il existe un vérifieur V

- *consistant* : si $x \in \mathcal{L}$, alors il existe un prouveur P tel que $Output(P, V)[x] = 1$.
- *significatif* : si $x \notin \mathcal{L}$, alors pour tout prouveur P', $Output(P', V)[x] = 0$.

L'extension suivante des protocoles interactifs devient naturelle :

Définition 12. $\mathcal{L} \in \mathcal{IP}$ s'il existe une machine de Turing polynomiale V

- *consistante* : si $x \in \mathcal{L}$, alors il existe un prouveur P tel que

$$\Pr[Output(P, V)[x] = 1] = 1.$$

- *significatif* : si $x \notin \mathcal{L}$, alors pour tout prouveur P', $\Pr[Output(P', V)[x] = 1] \leq 1/2$.

Avec ce qui précède, il est clair que $\mathcal{NP} \subset \mathcal{IP}$. En revanche, la réciproque ne semble pas vraie. En effet, on connaît des problèmes de $\mathcal{IP}$ que l'on n'a toujours pas prouvé être dans $\mathcal{NP}$ (le non isomorphisme de 2 graphes [46]). De plus, Shamir [103] a montré que $\mathcal{IP}$ est égal à $\mathcal{PSPACE}$, l'ensemble des langages qui peuvent être reconnus par des machines de Turing déterministes n'utilisant qu'un espace de travail (rubans) polynomial.

2.1.1.3 Preuves à divulgation nulle de connaissance

Dans la nouvelle définition interactive de $\mathcal{NP}$ (définition 11), on remarque qu'il existe toujours un prouveur convenable si $x \in \mathcal{L}$. En effet, il suffit de fournit un témoin w. Le vérifieur V peut alors faire tourner $M(x, w)$, où M est la machine de Turing de la première définition de $\mathcal{NP}$ (définition 9). Le problème est que le vérifieur apprend beaucoup plus de chose que la seule appartenance de x à $\mathcal{L}$. En d'autres termes, V sera capable, après cet entretien avec P, de convaincre un autre vérifieur V' que $x \in \mathcal{L}$. Cette preuve interactive de l'appartenance de x à $\mathcal{L}$ n'est donc pas à *divulgation nulle de connaissance*, elle apporte à V, non seulement la conviction que $x \in \mathcal{L}$, mais également des informations sur le moyen de prouver que $x \in \mathcal{L}$.

Définition 13. L'ensemble $\mathcal{ZKIP}$ est alors l'ensemble des problèmes de $\mathcal{IP}$ admettant des preuves interactives à *divulgation nulle de connaissance*.

Définissons plus rigoureusement la notion de *divulgation nulle de connaissance* :

Notation 14. Nous dénommerons $view(P, V)[x]$ tout ce que voit le vérifieur lors du protocole interactif (P, V) sur l'entrée x.

Remarque. Puisque P et V sont des machines probabilistes, dont l'exécution dépend des rubans aléatoires, $view(P, V)[x]$ définit une distribution de probabilités.

Définition 15. $\mathcal{L} \in \mathcal{ZKIP}$ si

- $\mathcal{L} \in \mathcal{IP}$;
- il existe un système interactif de preuve (P, V) pour $\mathcal{L}$ et une Machine de Turing Probabiliste Polynomiale $\mathcal{S}$ (appelée «simulateur») tels que pour tout $x \in \mathcal{L}$ et pour tout vérifieur Polynomial Probabiliste V', la distribution du ruban de sortie de $\mathcal{S}^{V'}[x]$ est indistinguable de $view(P, V')[x]$.

Remarque. Il existe plusieurs niveaux d'indistinguabilité, et donc de divulgation nulle de connaissance. En effet, deux distributions peuvent être égales, statistiquement indistinguables (la distance entre ces distributions est négligeable), polynomialement indistinguables (aucune machine de Turing polynomiale ne pourra distinguer les deux distributions) ou calculatoirement indistinguables (les distributions sont, en pratique, indistinguables).

Théorème 16 (Goldreich, Micali et Wigderson [47] et [121, 63]).
S'il existe des fonctions à sens-unique injectives alors $\mathcal{NP} \subset \mathcal{ZKIP}$.
Ainsi, tout problème de $\mathcal{NP}$ peut être prouvé de façon interactive et à divulgation nulle de connaissance (calculatoirement).

Théorème 17 (Ben-Or et al. [5] et [121, 63]).
S'il existe des fonctions à sens-unique injectives alors $\mathcal{IP} \subset \mathcal{ZKIP}$.
Ainsi, tout problème qui peut être prouvé interactivement peut l'être fait à divulgation nulle de connaissance (calculatoirement).

Des résultats plus récents [53, 55, 68] ont réduit les hypothèses aux fonctions à sens-unique non nécessairement injectives pour ce qui est des langages de $\mathcal{NP}$. Il semblerait qu'il ne soit pas possible de se passer de fonctions à sens-unique [79].

Remarque. Nous aurons également besoin dans la suite de **systèmes de preuve de connaissance à divulgation nulle face à un vérifieur honnête**. Un langage $\mathcal{L}$ peut être reconnu par un tel système si $\mathcal{L} \in \mathcal{IP}$, et s'il existe un système interactif de preuve (P, V) pour $\mathcal{L}$ et une Machine de Turing Probabiliste Polynomiale $\mathcal{S}$ (appelée «simulateur») tels que pour tout $x \in \mathcal{L}$, la distribution du ruban de sortie de $\mathcal{S}^V[x]$ est indistinguable de $view(P, V)[x]$. On remarque que dans ce cas, seule la distribution des communications avec le vérifieur V est simulable.

2.1.2 Les problèmes d'optimisation

2.1.2.1 Les classes d'optimisation

Certains problèmes, à défaut d'être résolus, pourraient être approximés. Par exemple, au lieu de chercher une table de vérité qui satisfait toutes les clauses d'un problème 3-SAT, on peut essayer de chercher une table de vérité qui satisfait le plus grand nombre de clauses. On appelle ce problème MAX-3-SAT. Ces problèmes d'optimisation sont également répartis en plusieurs classes :

- Les optimisations que l'on peut résoudre en temps polynomial.
- les optimisations que l'on peut approximer à tout facteur multiplicatif près en temps polynomial.

– Les optimisations que l'on ne peut approximer à mieux qu'un certain facteur multiplicatif près en temps polynomial, à moins que $\mathcal{P} = \mathcal{NP}$.

> ***Lemme 18 (Arora, Lund, Motwani, Sudan et Szegedy [1]).***
> *Il existe une constante $\varepsilon > 0$ telle que* Max-3-Sat *ne soit pas approximable à un facteur $1 + \varepsilon$ près (sous réserve que $\mathcal{P} \neq \mathcal{NP}$).*
> *Plus précisément, si $\mathcal{P} \neq \mathcal{NP}$ alors* Max-3-Sat *n'est pas approximable à mieux que $27/26$ près ($\varepsilon = 1/26$) [2].*

> ***Lemme 19 (Yannakakis [120]).***
> Max-3-Sat *est approximable à $1/7$ près.*

Exemple 5. Supposons que pour tout $\varepsilon > 0$, il existe un algorithme $Appr_\varepsilon$, polynomial, tel que pour tout problème 3-SAT, $\mathcal{C}$, de n clauses, dont $Opt(\mathcal{C})$ est le nombre maximal de clauses simultanément satisfaisables, $Opt(\mathcal{C})/(1+\varepsilon) \leq Appr_\varepsilon(\mathcal{C}) \leq Opt(\mathcal{C})$, alors $\mathcal{P} = \mathcal{NP}$.

– les optimisations que l'on ne peut approximer à mieux que n^ε près en temps polynomial.

Exemple 6. Supposons que pour tout $\varepsilon > 0$, il existe un algorithme $Appr_\varepsilon$, polynomial, tel que pour tout graphe G, de n sommets, dont la plus grande clique (sous-graphe complet) est de taille $Opt(G)$, $Opt(G)/n^\varepsilon \leq Appr_\varepsilon(G) \leq Opt(G)$, alors $\mathcal{P} = \mathcal{NP}$.

2.1.2.2 Réductions linéaires et Max-$\mathcal{SNP}$

Définition 20. Pour toute instance, le *coût* est une fonction qui à tout témoin potentiel associe un nombre. C'est cette fonction que l'on cherche à optimiser.

Par exemple, si $\mathcal{C}$ est une instance de Max-3-Sat, le coût d'un témoin (une distribution de vérité) est le nombre de clauses satisfaites.

Nous allons maintenant définir les réductions linéaires qui permettent de transformer un problème d'optimisation en un autre tout en conservant les mêmes propriétés d'approximation.

Définition 21 (L-réduction [80]). Soient deux problèmes d'optimisation Opt et Opt'. On dit que Opt se réduit linéairement à Opt' s'il existe deux algorithmes polynomiaux f et g et deux constantes $\alpha, \beta > 0$ tels que pour toute instance x de Opt :

1. L'algorithme f construit une instance x' de Opt' telle que $Opt'(x') \leq \alpha Opt(x)$.

2. Pour toute approximation y' de $Opt'(x')$, de coût $c'(y')$, l'algorithme g construit une approximation y de $Opt(x)$, de coût $c(y)$ telle que :

$$|Opt(x) - c(y)| \leq \beta |Opt'(x') - c'(y')|.$$

Définition 22 (MAX-$\mathcal{SNP}$). MAX-$\mathcal{SNP}$ est la classe des problèmes L-réductibles à MAX-3-SAT.

Définition 23 (MAX-$\mathcal{SNP}$-dur). Un problème d'optimisation est MAX-$\mathcal{SNP}$-dur si MAX-3-SAT s'y réduit linéairement.

Corollaire 24. *Propriétés :*

1. *Sous réserve que $\mathcal{P} \neq \mathcal{NP}$, pour tout problème d'optimisation* MAX-$\mathcal{SNP}$*-dur, il existe une constante pour laquelle ce problème n'est pas approximable.*
2. *Tout problème d'optimisation de* MAX-$\mathcal{SNP}$ *peut être approximé à une constante près.*

2.1.3 La sécurité et la théorie de la complexité

Tout au cours de cette thèse, nous allons démontrer la sécurité d'un certain nombre de schémas en allant de l'identification jusqu'à la monnaie électronique en passant par les signatures. Nos preuves de sécurité montreront l'inexistence de fraudeurs polynomiaux, donc vus comme des machines de Turing Polynomiales Probabilistes, avec une probabilité de succès non négligeable. Notre technique générale sera, pour cela, de transformer polynomialement une telle machine en une nouvelle machine de Turing Polynomiale capable de résoudre un problème reconnu difficile.

Parallèlement, nous traiterons un aspect plus pratique de la sécurité. C'est-à-dire que nous essaierons d'obtenir une réduction, la transformation polynomiale sus-citée, la plus efficace possible afin de s'approcher le plus possible de la notion de sécurité exacte [4].

2.2 Modèle de l'oracle aléatoire

Dans tout ce qui va suivre, nous nous placerons dans ce que Bellare et Rogaway [3] ont appelé le «modèle de l'oracle aléatoire». Ce modèle permet de supprimer l'obstacle des fonctions de hachage dans les preuves de sécurité. En effet, les seules propriétés de sens-unique et d'absence de collisions, qui sont des propriétés calculatoires, sont inutilisables en théorie de la complexité, qui traite de bornes asymptotiques. Nous considérerons donc les fonctions de hachage comme des fonctions parfaitement aléatoires, semblables à des oracles qui nous fournissent des valeurs aléatoires et indépendantes les unes des autres. Ce modèle n'est pas totalement nouveau, car il était plus ou moins supposé dans [41]. Mais Bellare et Rogaway ont formalisé cette notion assez récemment dans [3].

Définition 25. Pour toute constante k, un oracle aléatoire est une fonction f prise aléatoirement dans l'ensemble $\mathcal{F}_k$ des fonctions de $\{0,1\}^\star$ dans $\{0,1\}^k$.

Note 26. Ces fonctions vérifient les propriétés suivantes :

- Fonctions bornées : $(\forall x)(\forall f)(|f(x)| = k)$
- Fonctions aléatoires : la valeur en x est parfaitement indépendante des tirages précédents.

$$(\forall m \in \mathbb{N})$$
$$(\forall x_1, \ldots, x_m \in \{0,1\}^\star)(\forall y_1, \ldots, y_m \in \{0,1\}^k)$$
$$(\forall x \neq x_1, \ldots, x_m)(\forall y \in \{0,1\}^k)$$
$$\Pr_{f \in \mathcal{F}_k}\left[f(x) = y \;\middle|\; \begin{array}{c} f(x_1) = y_1 \\ \ldots \\ f(x_m) = y_m \end{array} \right] = \Pr_{f \in \mathcal{F}_k}[f(x) = y] = \frac{1}{2^k}.$$

Définition 27. Dans le «modèle de l'oracle aléatoire», on considérera qu'une fonction prise de façon aléatoire dans l'ensemble $\mathcal{F}_k$ est une bonne fonction de hachage. C'est-à-dire qu'on ne s'intéressera pas aux spécifications précises de la fonction de hachage et l'on se protégera ainsi contre les attaques génériques qui marchent indépendamment de ces spécifications.

Ainsi, dans le modèle de l'oracle aléatoire, une machine de Turing Polynomiale Probabiliste possède un ruban aléatoire et fait appel à un oracle aléatoire. Par conséquent, la probabilité de succès est évaluée sur l'espace produit des rubans aléatoires et des oracles aléatoires.

2.3 Machines de Turing à Oracle

Les participants ont bien entendu accès à la fonction de hachage qui est publique. Ils seront donc représentés par des machines de Turing Polynomiales Probabilistes à Oracle, $\mathcal{A}^f(\omega)$, qui

- font appel à l'oracle aléatoire f fournissant un condensé sur k bits, où k est polynomial en n, la taille de la donnée d'entrée.
- utilisent un ruban aléatoire ω.

Ces machines étant polynomiales, elles font un nombre polynomial d'appels distincts à l'oracle f : sur une entrée de taille n, on supposera que $\mathcal{A}^f(\omega)$ fait exactement Q appels distincts à l'oracle f (où Q est polynomial en n). En effet, si ce nombre d'appels devait être inférieur à Q, on modifie légèrement la machine afin qu'elle pose encore quelques nouvelles questions aléatoires à l'oracle pour atteindre ce nombre de Q. Les questions sont indicées par leur ordre chronologique, $\mathcal{Q}_i(\omega, f)$ pour $i = 1, \ldots, Q$.

Remarque. Les questions ne dépendent pas de tout f, mais simplement de la vue que $\mathcal{A}$ a de f, c'est-à-dire des réponses précédentes de l'oracle :

$$\text{pour tout } i,\ \mathcal{Q}_i(\omega, f) = \alpha_i\left(\omega, f(\mathcal{Q}_1), \ldots, f(\mathcal{Q}_{i-1})\right).$$

Ainsi, on peut remplacer l'oracle f par un oracle $\rho = (\rho_1, \ldots, \rho_Q)$ qui fournit successivement les valeurs ρ_1, ..., ρ_Q. Toutes les étapes de calcul, ainsi que les résultats des machines $\mathcal{A}^f(\omega)$ et $\mathcal{A}^\rho(\omega)$, où $f(\alpha_i(\omega, \rho_1, \ldots, \rho_{i-1})) = \rho_i$ pour tout $i \in \{1, \ldots, Q\}$, sont les mêmes. De plus, quel que soit R, $\Pr_{\omega,f}[\mathcal{A}^f(\omega) \to R] = \Pr_{\omega,\rho}[\mathcal{A}^\rho(\omega) \to R]$.

2.4 Un peu de probabilités

Dans nos preuves, nous utiliserons très souvent les lemmes suivants. Ils traduisent le fait que lorsqu'un ensemble A est «assez grand» dans un espace produit $X \times Y$, alors on peut séparer X en deux parties, X_0 et $\bar{X}_0$, telles que si $x \in X_0$, alors l'ensemble Y_x, des y vérifiant $(x, y) \in A$, est «assez grand» (voir figure 2.4).

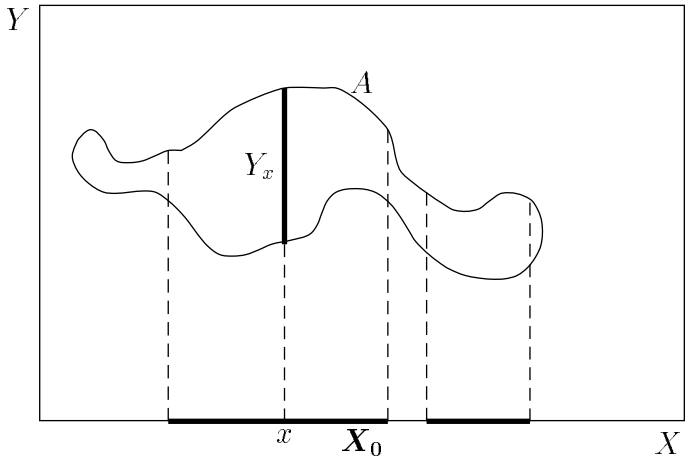

Fig. 2.4 – *Lemme de séparation*

Le premier lemme, le lemme de séparation 28, affirme que X_0 peut également être «assez grand». Quant au deuxième, le lemme de succès 29, il garantit qu'en cas de succès, $(x, y) \in A$, la probabilité pour que x soit dans X_0 est «assez grande».

Ces termes «assez grand» signifient que les sous-ensembles représentent une fraction non négligeable de l'ensemble complet.

Lemme 28 (Lemme de séparation).
Si A est un événement sur l'espace $X \times Y$ tel que $\Pr_{x,y}[A(x, y)] \geq \varepsilon$, alors, pour tout $\alpha < \varepsilon$, en notant

$$X_0 = \left\{ a \in X \;\middle|\; \Pr_y[A(x, y) | x = a] \geq \varepsilon - \alpha \right\},$$

i) $\Pr_x[x \in X_0] \geq \alpha$

ii) $(\forall a \in X_0) \quad \Pr_y[A(a, y)] \geq \varepsilon - \alpha.$

Preuve. Supposons $\Pr_x[x \in X_0] < \alpha$. Alors, en notant $\bar{X}_0 = X \setminus X_0$,

$$\begin{aligned}
\varepsilon &\leq \Pr_x[x \in X_0] \Pr[A(x,y) \big| x \in X_0] + \Pr_x[x \notin X_0] \Pr[A(x,y) \big| x \notin X_0] \\
\varepsilon &< \alpha \Pr[A(x,y) \big| x \in X_0] + \Pr_x[x \notin X_0] \sum_{a \in \bar{X}_0} \Pr_x[x = a \big| x \in \bar{X}_0] \Pr_y[A(x,y) \big| x = a] \\
\varepsilon &< \alpha + \Pr_x[x \notin X_0] \sum_{a \in \bar{X}_0} \Pr_x[x = a \big| x \in \bar{X}_0](\varepsilon - \alpha) \\
\varepsilon &< \alpha + \Pr_x[x \notin X_0](\varepsilon - \alpha) < \varepsilon
\end{aligned}$$

Ce qui entraîne une contradiction.

Par conséquent, nous avons bien $\Pr_{x \in X}[x \in X_0] \geq \alpha$. De plus, par définition, pour tout $a \in X_0$, $\Pr_y[A(a,y)] \geq \varepsilon - \alpha$. □

Ce lemme sera le plus souvent utilisé dans le cas particulier où $\alpha = \varepsilon/2$.

Lemme 29 (Lemme de succès).
Si A est un événement sur l'espace $X \times Y$ tel que $\Pr_{x,y}[A(x,y)] \geq \varepsilon$, alors, pour tout $\alpha < \varepsilon$, en notant

$$X_0 = \left\{ a \in X \;\middle|\; \Pr_y[A(x,y) \big| x = a] \geq \varepsilon - \alpha \right\},$$

on a

$$\Pr_{x,y}[x \in X_0 \,|\, A(x,y)] \geq \frac{\alpha}{\varepsilon}.$$

Preuve.

$$\begin{aligned}
\Pr_{x,y}[x \in X_0 \,|\, A(x,y)] &= \Pr_{x,y}[x \in X_0 \wedge A(x,y)] / \Pr_{x,y}[A(x,y)] \\
&= 1 - \left(\Pr_{x,y}[x \in \bar{X}_0 \wedge A(x,y)] \right) / \Pr_{x,y}[A(x,y)] \\
&= 1 - \left(\Pr_{x,y}[A(x,y) \,|\, x \in \bar{X}_0] \cdot \Pr_{x,y}[x \in \bar{X}_0] \right) / \Pr_{x,y}[A(x,y)] \\
&\geq 1 - \frac{(\varepsilon - \alpha)(1 - \alpha)}{\varepsilon} \\
&\geq 1 - \left(1 - \frac{\alpha}{\varepsilon}\right) \;\geq\; \frac{\alpha}{\varepsilon}
\end{aligned}$$

Ceci prouve le résultat. □

Chapitre 3

Identification

3.1 Les principes

L'identification est un outil très récent. Il s'impose dans tout système de communication à grande distance. Les protocoles utilisés sont très divers.

3.1.1 Le mot de passe, ou code secret

Le protocole d'identification, malheureusement le plus courant, est le contrôle par mot de passe ou PIN-code (Personal Identification Number). En effet, l'identification par mot de passe est présente sur les comptes UNIX ou autres systèmes d'exploitation multi-utilisateurs. Quant à l'utilisation du PIN-code, on la rencontre, en France, pour tout retrait d'argent, ou toute dépense par carte de crédit.

Description du mot de passe (sous UNIX)

- chaque utilisateur $\mathcal{U}$ choisit son mot de passe $x_{\mathcal{U}}$ qu'il envoie au serveur. Ce serveur le fait passer dans une fonction à sens-unique (type DES) : $y_{\mathcal{U}} = f(x_{\mathcal{U}})$. Il stocke alors les couples $(\mathcal{U}, y_{\mathcal{U}})$.
- lorsque $\mathcal{U}$ souhaite se connecter, il entre son login $\mathcal{U}$, puis son mot de passe x. Le serveur teste alors si $y_{\mathcal{U}} \stackrel{?}{=} f(x)$.

Pourquoi cette utilisation si courante est-elle malheureuse ? Un «simple» espionnage passif de la ligne (ligne téléphonique ou câble Ethernet dans un bâtiment) permet de récupérer le mot de passe qui y circule en clair.

En fait, il ne faudrait autoriser l'usage d'un mot de passe qu'une seule fois (one-time pad). Ceci nous amène alors à un protocole un peu plus évolué, appelé protocole de Lamport [61].

3.1.2 Le protocole de Lamport

Le protocole de Lamport [61] ressemble un peu au mot de passe, mais est beaucoup plus sûr. Le mot de passe est renouvelé à chaque utilisation. En revanche, il nécessite une certaine capacité de calcul de la part du prouveur, ou à défaut, une capacité de stockage.

Description du protocole de Lamport

- Initialisation
  - chaque utilisateur $\mathcal{U}$ choisit et stocke un nombre secret $x_{\mathcal{U}} = x_0$.
  - ensuite, il calcule $x_1 = f(x_0)$, $x_2 = f(x_1)$, ..., $x_{1000} = f(x_{999})$, où f est une fonction à sens unique.
  - Il communique alors $y_{\mathcal{U}} = x_{1000}$ au vérifieur qui stocke le couple $(\mathcal{U}, y_{\mathcal{U}})$.
- Identification
  - Lors du premier contrôle d'accès, le prouveur envoie $\mathcal{U}$ et $x = x_{999}$. Le vérifieur peut contrôler que $f(x) = y_{\mathcal{U}}$, et remplace $y_{\mathcal{U}}$ par x dans le couple de $\mathcal{U}$.
  - Au prochain contrôle d'accès, le prouveur enverra $x = x_{998}$, puis x_{997}, etc. Seul le vrai prouveur peut calculer les antécédents successifs de x_{1000}.

Lors d'un crash du disque, les valeurs des précédents contrôles d'accès sont détruites, mais peuvent avoir été notées par un imposteur. Supposons qu'aucune sauvegarde des disques n'ait été effectuée depuis le 100$^{\text{e}}$ contrôle d'accès (*i.e.* la sauvegarde contient le couple $(\mathcal{U}, x_{900})$). Au prochain contrôle, le vérifieur réclamera x_{899} qui n'est plus secret car a déjà été dévoilé.

Le deuxième inconvénient, et sans doute le plus important pour un grand nombre d'applications, telles que le retrait d'argent ou les achats par carte de crédit, est que cette identification ne peut s'effectuer qu'auprès d'un unique vérifieur.

3.1.3 Preuves interactives

On peut utiliser les preuves interactives pour effectuer des identifications. Le but du prouveur est alors de convaincre le vérifieur qu'il connaît un secret S (qu'il est normalement le seul à connaître).

Un protocole interactif d'identification met en scène le prouveur Alice et le vérifieur Bob. Alice veut prouver interactivement son identité à Bob. Pour cela, Alice possède une clé publique $I_{\mathcal{A}}$, et une clé secrète $S_{\mathcal{A}}$. La clé publique $I_{\mathcal{A}}$ est connue de tous, et il n'y a aucun doute sur l'identité de son propriétaire. Elle représente un mot d'un langage $\mathcal{L} \in \mathcal{NP}$. La clé secrète $S_{\mathcal{A}}$ est un témoin de l'appartenance de $I_{\mathcal{A}}$ à $\mathcal{L}$. On choisit le langage $\mathcal{L}$ de telle sorte que trouver un tel témoin est difficile.

La preuve d'identité consiste pour Alice à prouver qu'elle connaît un témoin de l'appartenance de $I_{\mathcal{A}}$ à $\mathcal{L}$. Pour tout autre individu, une tentative de se faire passer pour Alice échouera s'il ne connaît pas de témoin.

Divulgation nulle de connaissance. La théorie du *zero-knowledge* [48] ou protocoles à divulgation nulle de connaissance a montré qu'Alice pouvait prouver sa connaissance d'un tel témoin sans révéler ce témoin, ni même une information partielle, contrairement au système du mot de passe.

Les premiers protocoles *zero-knowledge* furent basés sur des problèmes de théorie des nombres (Fiat-Shamir [41] et ses variantes [51, 52, 78, 74], Schnorr [98, 99], etc). Ils ont deux inconvénients majeurs :

- les problèmes utilisés (factorisation, problème RSA et logarithme discret) ne sont pas $\mathcal{NP}$-complets et des algorithmes, des nouvelles techniques [107, 108] ainsi que des ordinateurs de plus en plus performants les menacent.
- les opérations nécessaires (multiplications et exponentiations modulaires) sont très coûteuses en temps et en puissance machine.

En revanche, depuis 1989, de nouveaux schémas sont apparus. Ils reposent sur des problèmes $\mathcal{NP}$-complets, et ne nécessitent des opérations que sur des petits entiers, voire sur des bits : PKP (Permuted Kernel Problem) [102], SD (Syndrome Decoding) [112], CLE (Constrained Linear Equations) [113] et tout dernièrement PPP (Permuted Perceptrons Problem) [83, 84].

Après cette introduction générale sur les schémas d'identification, nous présenterons, dans le chapitre 4, le tout dernier de ces schémas, basé sur le Problème des Perceptrons Permutés.

Protocoles basés sur l'identité. Le problème posé par ces protocoles à clés publiques est la diffusion des clés publiques. En effet, il faut être sûr que la clé publique que l'on utilise appartient bien à la personne avec laquelle on veut communiquer. Si Charlie remplace la clé publique d'Alice par sa propre clé publique, il n'aura aucun mal à se faire passer pour Alice auprès de Bob. Il faut donc que les clés publiques soient certifiées. C'est-à-dire qu'une autorité digne de confiance signe les couples (identifiant, clé publique), ainsi ce couple ne pourra être modifié.

Une autre possibilité est d'utiliser des protocoles basés sur l'identité. Dans ce cas, la clé publique est tout simplement l'identité de l'individu.

3.2 Sécurité

La sécurité d'un schéma d'identification consiste en la garantie pour Alice que personne ne pourra se faire passer pour elle, quels que soient les moyens techniques employés par le fraudeur. Deux types de fraudes sont usuellement considérés : les attaques passives et les attaques actives. Bien évidemment, l'attaque active sera la plus puissante, nous nous attacherons donc par la suite à prouver la sécurité des schémas d'identification face à ce type d'attaque. Ainsi, nous appellerons un schéma **sûr**, un schéma tel qu'un fraudeur ne puisse se faire passer pour Alice qu'avec probabilité négligeable, même suite à une attaque active.

3.2.1 Attaques passives

Une attaque passive consiste en l'observation «muette» de la communication qu'Alice entretient avec Bob pour lui prouver son identité. L'attaquant tente, après une telle observation, de se faire passer pour Alice auprès de Bob. Le système du mot de passe, ou

plus généralement de l'identification présentée figure 3.1, est très vulnérable à ce type d'attaque.

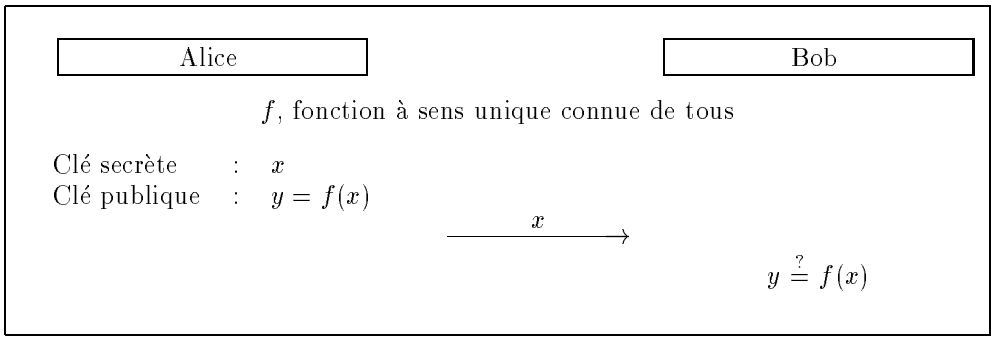

Fig. 3.1 – *Mot de passe*

Il serait donc souhaitable que ce type d'attaque, que quiconque peut aisément mettre en œuvre, n'affecte pas la sécurité du protocole. La première règle est donc d'éviter le rejeu : ce qu'envoie le prouveur doit être différent à chaque preuve d'identité. Pour cela, la preuve se transforme en une série de questions-réponses : Bob pose les questions et Alice y répond. Ce système interactif de preuve doit être :

consistant : Alice qui répond honnêtement aux réponses doit être acceptée.

significatif : Si Charlie essaie de se faire passer pour Alice, il aura très peu de chance d'être accepté.

3.2.2 Attaques actives

Un deuxième type d'attaque devient désormais envisageable. Supposons que Charlie prenne la place du vérifieur, il peut poser des questions de façon à obtenir une information sur la clé secrète d'Alice. Il peut ensuite tenter de se faire passer pour Alice auprès de Bob. Charlie est, cette fois-ci, actif dans l'attaque.

Pour contrecarrer ce genre d'attaque, plusieurs types de preuves de connaissance sont apparus.

3.2.2.1 Preuves interactives à divulgation nulle de connaissance

Dans une preuve interactive à divulgation nulle de connaissance, la communication entre le prouveur, Alice, et tout vérifieur, honnête comme Bob, ou même malhonnête comme Charlie, possède la propriété d'être simulable par une machine de Turing Polynomiale Probabiliste $\mathcal{S}$ qui ne connaît pas la clé secrète d'Alice.

Cette propriété est fondamentale contre les attaques actives. Quel que soit le vérifieur, l'information effectivement transmise par Alice est uniquement la connaissance d'un témoin. Aucune information, même partielle, sur ce témoin, en l'occurrence la clé secrète, n'est fournie au vérifieur ou à tout espion. Plutôt que de longues explications, l'exemple suivant va aider à comprendre le lien entre ces preuves et les protocoles d'identification.

Voici donc le tout premier schéma d'identification interactif à divulgation nulle de connaissance présenté en 1986 par Amos Fiat et Adi Shamir [41] (voir figure 3.2) contre lequel une attaque active est équivalente à la factorisation.

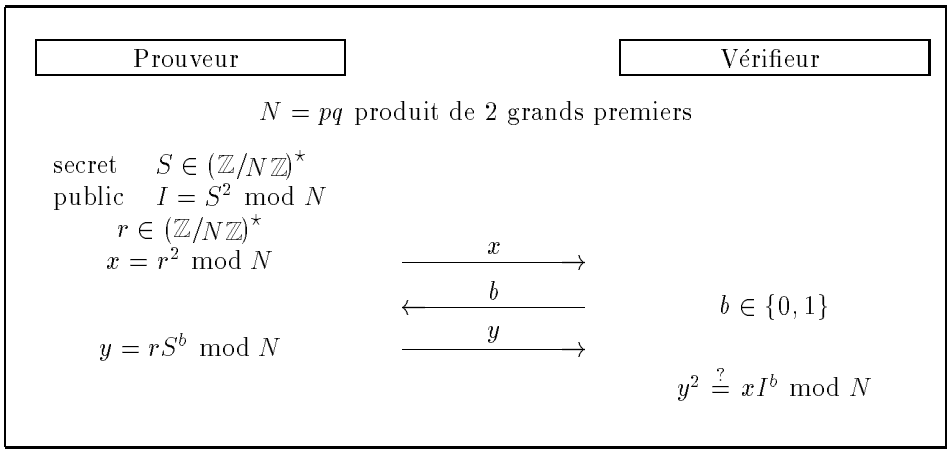

Fig. 3.2 – *Schéma de Fiat-Shamir à un secret*

- Initialisation
  - Une autorité choisit un grand nombre composé $N = pq$, de taille n, et le publie.
  - Alice choisit une clé secrète $S \in (\mathbb{Z}/N\mathbb{Z})^\star$, puis calcule $I = S^2 \bmod N$. Elle publie alors cette clé publique I.
- Identification
  1. Alice choisit un nombre aléatoire $r \in_R (\mathbb{Z}/N\mathbb{Z})^\star$, elle calcule $x = r^2 \bmod N$ qu'elle envoie à Bob.
  2. Bob choisit un bit b aléatoire, puis l'envoie à Alice.
  3. Alice envoie alors $y = r \cdot S^b \bmod N$ à Bob.
  4. Bob vérifie si $y^2 = x \cdot I^b \bmod N$.

  Ce protocole est répété k fois séquentiellement, où k est le facteur de sécurité. Pour obtenir une sécurité super-polynomiale, c'est-à-dire telle que la probabilité de fraude soit inférieure à l'inverse de tout polynôme en n, il faut choisir $k \gg \log n$ tout en restant polynomial en n, où $n = \log N$ est la taille des clés.

Il est important de noter que ces répétitions doivent être effectuées séquentiellement et non en parallèle si on veut conserver la propriété de divulgation nulle de connaissance. Dans le cas contraire, la simulation polynomiale devient impossible.

Nous verrons au paragraphe suivant que pour ce qui est du schéma de Fiat-Shamir, la répétition parallèle permet tout de même de garantir la sécurité face aux attaques actives. Ceci provient du fait que ce protocole est de plus à «témoin indistinguable».

Lemme 30 .
Le schéma d'identification de Fiat-Shamir séquentiel est un système interactif de preuve de connaissance d'une racine carrée.

Preuve. Pour cela, il nous faut montrer que ce protocole est :

consistant : Alice est capable, à r fixé, de répondre aux deux questions ($b = 0$ et $b = 1$).

significatif : Réciproquement, supposons que Charlie, noté $\mathcal{C}$, soit capable de répondre correctement avec probabilité supérieure à $1/2+1/poly(n)$. Il existe alors un extracteur utilisant $\mathcal{C}$ capable de calculer une racine carrée de I. En effet, sur une fraction non négligeable d'engagements x, $\mathcal{C}$ sait répondre aux deux questions ($b = 0$ et $b = 1$). En réitérant ces tentatives un nombre polynomial de fois on obtient un tel engagement de la part de $\mathcal{C}$, on peut alors lui poser les deux questions (avec un reset de son ruban). Il nous retourne alors y_0 et y_1 tels que $y_0^2 = x \bmod N$ et $y_1^2 = x \cdot I \bmod N$. Par conséquent, si nous posons $z = y_1/y_0 \bmod N$, nous obtenons une racine carrée de I, car $I = z^2 \bmod N$.

Charlie a donc au plus une chance sur deux d'être accepté à chaque itération. De la même manière, nous pouvons montrer qu'après k tours successifs, la probabilité de succès d'un fraudeur est inférieure à 2^{-k}. Après simplement 20 itérations, cette probabilité est alors inférieure à un pour un million. □

Lemme 31 .
Le schéma d'identification de Fiat-Shamir séquentiel est à divulgation nulle de connaissance.

Preuve. Nous avons vu que pour être à divulgation nulle de connaissance, un protocole doit être simulable. Nous allons donc construire un simulateur $\mathcal{S}$ capable de fabriquer des rubans de communication avec une distribution indistinguable de celle des rubans obtenus lors de communications entre Alice et Charlie.

Charlie est un vérifieur malhonnête, donc il ne pose pas obligatoirement la question b de façon aléatoire, comme il le devrait. Il possède une stratégie, éventuellement probabiliste, qui dépend de x, $b = f(x)$.

1. $\mathcal{S}$ tente de deviner la question que Charlie va lui poser : $c \in \{0, 1\}$.

2. $\mathcal{S}$ choisit aléatoirement $y \in (\mathbb{Z}/N\mathbb{Z})^\star$ et calcule $x = y^2/I^c \bmod N$.
   On peut remarquer que x est alors un résidu quadratique uniformément distribué.

3. $\mathcal{S}$ interroge alors la stratégie du vérifieur, $b = f(x)$.

4. Si $b = c$ alors on écrit x, b et y sur le ruban, sinon, on réinitialise le vérifieur au début de ce tour, et on retourne en 1.

La simulation de k tours successifs se fait donc en moyenne en $\mathcal{O}(k)$. En revanche, la simulation du protocole parallèle nécessiterait en moyenne $\mathcal{O}(2^k)$ itérations. Ce qui explique que la version parallèle n'est plus à divulgation nulle de connaissance. □

Le résultat de sécurité suivant en découle :

Théorème 32 .
Une attaque active contre le schéma d'identification de Fiat-Shamir séquentiel est équivalente à la racine carrée.

Preuve. Soit un élément résidu quadratique I de $(\mathbb{Z}/N\mathbb{Z})^\star$. On considère cet élément comme la clé publique d'Alice.

Supposons que Charlie soit un attaquant capable de se faire passer pour elle avec probabilité non négligeable $\varepsilon > 1/P$, où P est un polynôme en n ; et cela, après avoir servi Q fois de vérifieur à Alice.

Nous pouvons donc effectuer la simulation du ruban de communication que Charlie obtiendrait avec Alice. Cette simulation prend un temps moyen en $\mathcal{O}(kQ)$, polynomial en n. Ensuite, avec l'information accumulée, Charlie parvient à franchir l'identification avec probabilité non négligeable $\varepsilon \geq 1/P$. Comme $k \gg \log n$, pour n assez grand, nous avons $2^k \geq 2P$. Cela signifie que dans l'arbre T des exécutions gagnantes de profondeur k, il existe des nœuds admettant deux fils. De plus, il est aisé de vérifier à l'aide d'une démonstration semblable au lemme de succès 29 que parmi ces exécutions gagnantes, une sur deux contient un tel nœud. Faisons alors tourner l'attaque jusqu'à l'obtention d'un succès. Avec probabilité un demi, nous passons par un tel nœud. Rejouons dans les mêmes conditions (mêmes rubans aléatoires, etc) en changeant seulement la question d'indice β. Cet indice est choisi au hasard, alors avec probabilité non négligeable (supérieur à $1/k$) cet indice pointe sur le nœud à deux fils. Dans ce cas, Charlie sait également répondre à cette deuxième question. Alors, comme pour montrer que le schéma est significatif, on extrait de Charlie une racine carrée de I. □

Remarque. On peut bien évidemment prouver l'équivalence entre la racine carrée et la factorisation : Soit S un élément quelconque de $(\mathbb{Z}/N\mathbb{Z})^\star$. On calcule son carré $I = S^2 \bmod N$. On fait alors tourner la simulation puis l'attaque précédente qui fournit une racine carrée S' de I. Cette racine provient d'une simulation et d'une attaque indépendantes de S. Par conséquent, avec probabilité supérieure à $1/2$, le plus grand diviseur commun de N et de $S' - S$ fournit un facteur de N.

Un autre schéma d'identification à divulgation nulle de connaissance bien connu est celui de Schnorr [98, 99] (voir figure 3.3) basé, cette fois-ci, sur le problème du logarithme discret dans un sous-groupe premier.

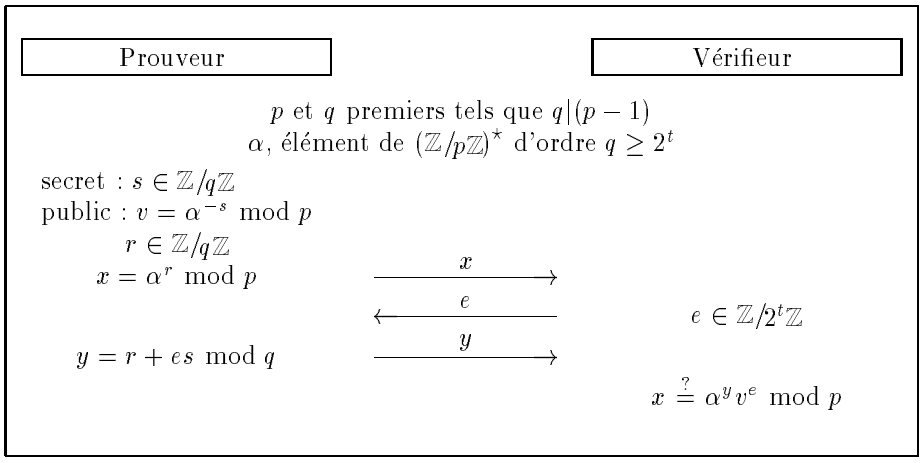

Fig. 3.3 – *Schéma d'identification de Schnorr*

Lemme 33 .
Le schéma d'identification de Schnorr est un système interactif de preuve de connaissance du logarithme discret.

Preuve. Pour cela, il nous faut montrer que ce protocole est :

consistant : Alice est capable, à r fixé, de répondre aux 2^t questions ($e \in \mathbb{Z}/2^t\mathbb{Z}$).

significatif : Réciproquement, une personne capable de répondre, pour l'engagement x effectué, à deux questions ($e_0 \neq e_1 \in \mathbb{Z}/2^t\mathbb{Z}$), connaît y_0 et y_1 tels que

$$x = \alpha^{y_0} v^{e_0} = \alpha^{y_1} v^{e_1} \mod p.$$

Par conséquent, $v^{e_1 - e_0} = \alpha^{y_0 - y_1} \mod p$. Si nous posons $z = (y_0 - y_1)/(e_1 - e_0) \mod q$, alors z est le logarithme discret de v relativement à α.

Charlie, qui ignore la clé secrète d'Alice, ne pourra pas répondre à plus d'une question. Sa probabilité de fraude est donc inférieure à $1/2^t$. □

Une technique semblable à celle utilisée pour Fiat-Shamir permet de montrer que ce schéma, pour t fixé, est à divulgation nulle de connaissance. Il n'est, en revanche, pas conseillé de faire augmenter la taille de t pour augmenter la sécurité du protocole. En effet, pour $t \gg \log n$, ce qui est nécessaire pour obtenir une sécurité super-polynomiale, la simulation ne peut plus être polynomiale. La protocole n'est alors plus prouvé résistant

aux attaques actives. Comme pour le schéma de Fiat-Shamir, une sécurité élevée s'obtient en réitérant le schéma de base.

Ces deux schémas reposent sur des problèmes de théorie des nombres. Il en existe de plus récents basés sur des problèmes combinatoires $\mathcal{NP}$-complets. Le premier de ce type fut le schéma de Shamir [102] basé sur PKP. Ensuite, Jacques Stern en a proposé un sur le problème du décodage par syndrome [112], puis un sur le problème des systèmes d'équations linéaires contraintes [113]. Enfin, le plus récent fut présenté par l'auteur de cette thèse en 1995 à Eurocrypt '95 [83], et sera analysé en détail au chapitre 4.

On remarque cependant que ces schémas à divulgation nulle de connaissance, pour résister aux attaques actives, sont très coûteux. Ils nécessitent un nombre important d'interactions [56]. En 1990, Feige et Shamir [38] ont présenté les notions de protocoles *witness-hiding* (ou à «témoin caché») et *witness-indistinguishable* (ou à «témoin indistinguable»). Ces notions permettent à certains protocoles d'être sûrs contre les attaques actives, sans être à divulgation nulle de connaissance, tout en ne nécessitant que 3 passes.

3.2.2.2 Témoin indistinguable

Nous ne nous intéresserons qu'aux schémas à «témoin indistinguable». Pour ces schémas, il existe plusieurs clés secrètes associées à une même clé publique. Mais la vue du vérifieur est indépendante de la clé secrète utilisée par le prouveur.

Le tout premier exemple est en fait le schéma de Fiat-Shamir et ses variantes. En effet, à un résidu quadratique sont associées quatre racines carrées (si on utilise un module RSA N, à deux facteurs premiers). Deux quelconques de ces racines fournissent, avec probabilité un demi, un facteur non trivial de N. En outre, toutes ces racines produisent des rubans de communications suivant des distributions indistinguables. Ainsi, il n'est plus utile de pouvoir simuler une interaction avec un vérifieur, même malhonnête, sans clé secrète. En effet, nous pouvons transformer une attaque active en une machine capable de factoriser les modules RSA : soit N à factoriser.

- On choisit un élément aléatoire $S \in (\mathbb{Z}/N\mathbb{Z})^\star$.
- On calcule son carré, $I = S^2 \bmod N$.
- On effectue alors un certain nombre d'identifications auprès de Charlie (vérifieur malhonnête).
- Après un nombre polynomial d'identifications, Charlie est capable de se faire passer pour Alice avec probabilité non négligeable. On parvient alors, à l'aide d'un rejeu comme utilisé précédemment, à extraire une racine carrée S' de I. Or, lors des interactions, l'information qu'il a pu accumuler est indépendante de la racine de I utilisée par le prouveur. Alors, avec probabilité $1/2$, S et S' permettent d'obtenir un facteur non trivial de N.

Théorème 34 .
Une attaque active contre le schéma d'identification de Fiat-Shamir parallèle est équivalente à la factorisation.

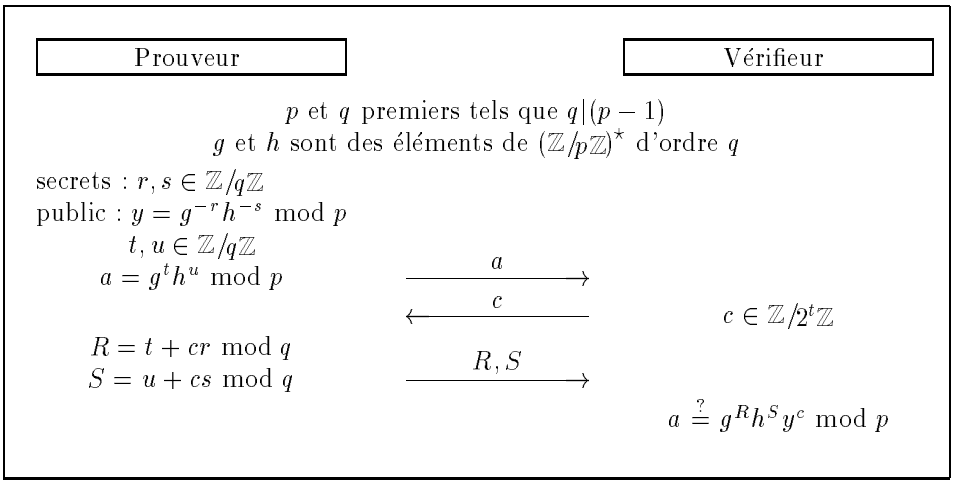

Fig. 3.4 – *Adaptation «à témoin indistinguable» du schéma de Schnorr*

Okamoto [77], en 1992, a présenté deux adaptations à témoin indistinguable des schémas de Schnorr [98, 99] et de Guillou-Quisquater [51, 52]. La première est bien entendu relative au logarithme discret (voir figure 3.4), la seconde repose sur le problème RSA (voir figure 3.5).

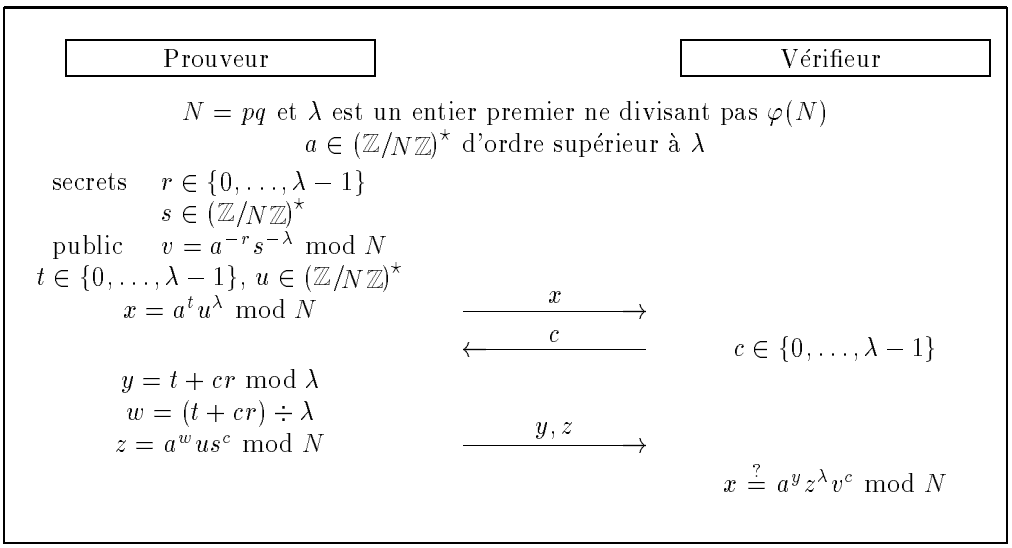

Fig. 3.5 – *Adaptation «à témoin indistinguable» du schéma de Guillou-Quisquater*

En effet, pour le premier, la connaissance de deux témoins distincts (r_1, s_1), (r_2, s_2), associés à une même clé publique y, $y = g^{-r_1}h^{-s_1} = g^{-r_2}h^{-s_2} \bmod p$, fournit le logarithme discret de h relativement à g : posons $\ell = (s_2 - s_1)/(r_1 - r_2) \bmod q$, alors $h = g^\ell \bmod p$.

Quant à la variante de Guillou-Quisquater, la connaissance de deux témoins distincts (r_1, s_1), (r_2, s_2), associés à une même clé publique v, $v = a^{-r_1}{s_1}^\lambda = a^{-r_2}{s_2}^\lambda \bmod n$ fournit $a^{r_1 - r_2} = (s_2/s_1)^\lambda \bmod n$. Posons $b = s_2/s_1 \bmod n$. Comme $r_1 - r_2$ est premier avec λ, il existe, d'après l'égalité de Bezout, deux entiers u et v tels que $u(r_1 - r_2) + v\lambda = 1$. Alors

on a $(a^{r_1-r_2})^u = a^{1-v\lambda} = b^{u\lambda} \bmod n$. Ainsi, $a = (b^u a^v)^\lambda$, ce qui fournit une racine λ-ième de a et contredit l'hypothèse RSA.

Ces schémas à «témoin indistinguable» fournissent donc des protocoles d'identification sûrs face à des attaques actives en simplement trois passes. Ce qui n'est pas possible avec des protocoles à divulgation nulle de connaissance [56].

Ils auront de plus un grand intérêt lors de l'étude des signatures en blanc au chapitre 6.

3.2.2.3 Nouveau «design»

À Eurocrypt '96, Shoup [109] a présenté une preuve originale de la sécurité d'un cas particulier du schéma de Ong-Schnorr [78]. Sans être à «divulgation nulle de connaissance», ni à «témoin indistinguable», ce schéma (voir figure 3.6) est prouvé sûr contre les attaques actives.

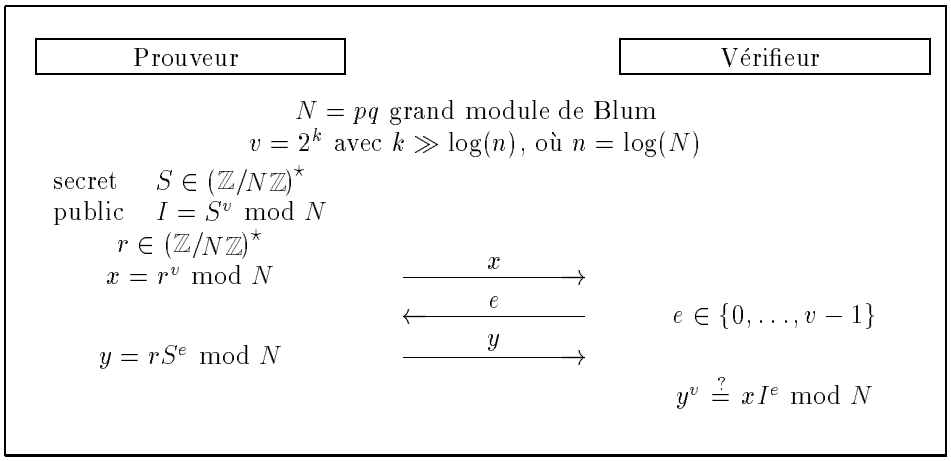

Fig. 3.6 – *Schéma d'identification de Ong-Schnorr*

Sa preuve utilise la propriété des entiers de Blum. En effet, ces entiers de la forme $N = pq$ où p et q sont congrus à 3 modulo 4 ont la particularité de faire de la fonction carrée modulo N une bijection de l'ensemble des résidus quadratiques dans lui-même. Schnorr [100] a généralisé ce résultat dans diverses directions.

Pour obtenir une sécurité super-polynomiale, dans ce protocole, il faut prendre v plus grand que tout polynôme en n, où n est la taille de N, le module. Cela se traduit par $k \gg \log(n)$.

Théorème 35 .
Une attaque active contre le schéma d'identification de Ong-Schnorr (figure 3.6) est équivalente à la factorisation.

Preuve. Soit $\mathcal{C}$ un attaquant capable de se faire passer pour Alice avec probabilité non négligeable $\varepsilon \geq 1/P$, après avoir interagi Q fois avec elle en tant que vérifieur (malhonnête), où P et Q sont deux polynômes en n. Posons $\lambda = \lceil \log P \rceil + 1$ puis $w = 2^\lambda$. Alors,

$4P > 2^\lambda = w \geq 2P$. Pour n assez grand, $v = 2^k > 4P > 2^\lambda = w \geq 2P$. Une simulation polynomiale indistinguable est alors possible (voir figure 3.7), sa probabilité de succès est de $2^{-\lambda}$, supérieure à $1/4P$. En effet, elle utilise une pseudo-clé secrète S qui est, non pas la

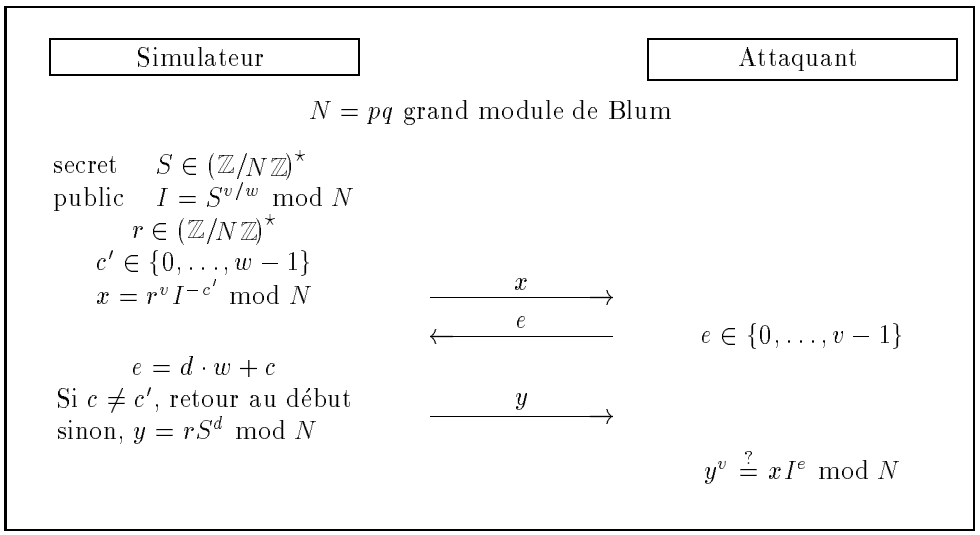

Fig. 3.7 – *Simulation du schéma d'identification de Ong-Schnorr*

racine v-ième de I, mais sa racine v/w-ième, où w dépend de la probabilité de succès de l'attaquant. Il tente de deviner partiellement le challenge qui lui sera posé. La probabilité de trouver le challenge entier est négligeable en raison du caractère super-polynomial de v. En revanche, deviner le challenge modulo w réussit avec probabilité $1/w$, supérieure à $1/4P$.

Ainsi, on simule Q fois la preuve de connaissance devant l'attaquant, pour lui permettre d'accumuler de l'information Inf. Ces Q simulations nécessitent en moyenne $Qw < 4PQ$ itérations. Puis on le laisse s'identifier : $\Pr_{\omega,e}[\mathcal{C}^{Inf} \text{ succès}] \geq \varepsilon \geq 1/P \geq 2/w$, où ω est le ruban aléatoire de $\mathcal{C}$, et détermine entièrement ce qu'il envoie au premier tour, x. D'après le lemme de séparation 28, il existe un ensemble Ω tel que

i) $\Pr[\omega \in \Omega] \geq 1/2w$

ii) pour tout $\omega \in \Omega$, $\Pr_e[\mathcal{C}^{Inf} \text{ succès}] \geq 3/2w$.

Choisissons (ω, e) au hasard. D'après le lemme de succès 29, avec probabilité supérieure à $1/2w \geq 1/8P$, $\omega \in \Omega$ et l'exécution aboutit à un succès avec une mise en gage initiale x. Comme cette valeur initiale ne dépend que de ω, relancer l'attaque avec un même ruban aléatoire ω fournit le même x.

$$\begin{aligned}
\Pr_{e'}\left[\begin{array}{c}\mathcal{C}^{Inf} \text{ succès} \\ e' \neq e \bmod w\end{array}\right] &= \Pr_{e'}[\mathcal{C}^{Inf} \text{ succès}] - \Pr_{e'}[\mathcal{C}^{Inf} \text{ succès et } e' = e \bmod w] \\
&\geq \Pr_{e'}[\mathcal{C}^{Inf} \text{ succès}] - \Pr_{e'}[e' = e \bmod w] \geq \frac{3}{2w} - \frac{1}{w} \geq \frac{1}{2w} > \frac{1}{8P}.
\end{aligned}$$

Ainsi, en réitérant suffisamment de fois l'attaque avec le même ω et un challenge e'

aléatoire, on obtient un nouveau succès avec $e \neq e' \bmod w$. Il vérifie

$$x = y^v I^{-e} = y'^v I^{-e'} \bmod N.$$

Posons $f = e' - e = 2^t \cdot u$, avec u impair. Puisque $e' - e \neq 0 \bmod 2^\lambda$, on a $0 \leq t < \lambda$. On pose également $z = (y'/y)^{2^{\lambda - t}} \bmod N$, qui est toujours un résidu quadratique. Alors, $(z^{v/w})^{2^t} = z^{2^{k-\lambda+t}} = (I^u)^{2^t} \bmod N$. Or, l'application carré est une bijection de l'ensemble des résidus quadratiques dans lui-même lorsque N est un entier de Blum. Par suite, $z^{v/w} = I^u \bmod N$. Cependant, par définition, $I^u = (S^u)^{v/w} \bmod N$. Par transitivité, il vient $z^{v/w} = (S^u)^{v/w} \bmod N$. En utilisant encore une fois la bijectivité de la fonction carré, nous obtenons l'égalité $z^2 = (S^u)^2 \bmod N$. Comme u est impair, S^u appartient à la même classe de résiduosité quadratique que S. Quant à z, c'est nécessairement un carré. Par conséquent, avec probabilité un demi, le pgcd de $z - S^u$ avec N fournit un facteur non trivial de N.

Nous avons donc transformé une machine de Turing Polynomiale capable de s'identifier avec probabilité asymptotiquement non négligeable selon une attaque active en une machine de Turing Polynomiale capable de factoriser les entiers de Blum avec probabilité non négligeable.

Il faut remarquer que cette transformation est non uniforme. C'est-à-dire que la longueur des boucles itératives et certains paramètres de la transformation dépendent de la probabilité de succès de l'attaquant. □

Chapitre 4

Les perceptrons

Après avoir expliqué en quoi consiste un protocole d'identification, et quelles propriétés de sécurité il doit satisfaire, nous allons en étudier un tout nouveau, présenté par l'auteur de cette thèse à Eurocrypt '95 [83].

Ce protocole fait partie de la famille des protocoles d'identification à divulgation nulle de connaissance qui reposent sur un problème $\mathcal{NP}$-complet. Le problème de base est le problème des perceptrons permutés (ou **PPP**).

Nous allons tout d'abord présenter le problème, avec ses propriétés. Nous allons ensuite étudier un certain nombre d'attaques afin d'évaluer sa difficulté en moyenne. Puis, nous allons l'utiliser à des fins cryptographiques dans un protocole d'identification à divulgation nulle de connaissance. Nous terminerons par les performances pratiques obtenues grâce à une implantation sur carte à microprocesseur.

4.1 Le problème des perceptrons

4.1.1 Énoncé du problème

Le problème suivant apparaît en physique lors de l'étude des perceptrons d'Ising, ainsi qu'en intelligence artificielle avec les réseaux de neurones. Nous le dénommerons *Problème des Perceptrons*.

Définition 36. Nous appellerons ε-matrice (resp. ε-vecteur) une matrice (resp. ε-vecteur) dont les composantes sont $+1$ ou -1.

Problème 37 (Problème de décision – PP).
Donnée : Une ε-matrice A de taille $m \times n$.
Question : Existe-t-il un ε-vecteur Y de taille n tel que $A \cdot Y \geq 0$?

Problème 38 (Problème de recherche).
Donnée : Une ε-matrice A de taille $m \times n$.
Question : Trouver un ε-vecteur Y de taille n tel que $A \cdot Y \geq 0$.

4.1.2 $\mathcal{NP}$-complétude

Théorème 39 .
Le Problème des Perceptrons est $\mathcal{NP}$-complet.

Preuve. Montrons donc que ce problème est dans $\mathcal{NP}$, puis qu'il est $\mathcal{NP}$-dur.

(1) Tout d'abord, il est clair que ce problème est dans $\mathcal{NP}$. Une solution au problème de recherche fournit un témoin idéal.

(2) Ensuite, montrons que ce problème est $\mathcal{NP}$-dur. Pour cela, nous allons réduire de façon polynomiale une instance de 3-SAT à une instance de ce problème. Soit donc $\mathcal{C} = \{C^1, \ldots, C^q\}$ une instance de 3-SAT. Chaque clause C^i est une disjonction de trois littéraux sur les variables x_1, ..., x_k. Nous la noterons $C^i = \{\ell_1^i, \ell_2^i, \ell_3^i\}$ avec $\ell_j^i \in \{x_1, \bar{x}_1, \ldots, x_k, \bar{x}_k\}$.

À chaque clause C^i, on associe le ε-vecteur X^i de taille $2k$:

$$\begin{array}{lllllllll} \forall p \in \{1,\ldots,k\} & \text{si } x_p \in C^i, & X_p^i & = & +1 & \text{et} & X_{p+k}^i & = & +1, \\ & \text{si } \bar{x}_p \in C^i, & X_p^i & = & -1 & \text{et} & X_{p+k}^i & = & -1, \\ & \text{sinon} & X_p^i & = & +1 & \text{et} & X_{p+k}^i & = & -1. \end{array}$$

À une distribution de vérité D, on associe le ε-vecteur V de taille $2k$:

$$\begin{array}{lllllllll} \forall p \in \{1,\ldots,k\} & \text{si } x_p \in D, & V_p & = & +1 & \text{et} & V_{p+k} & = & +1, \\ & \text{si } \bar{x}_p \in D, & V_p & = & -1 & \text{et} & V_{p+k} & = & -1. \end{array}$$

Or un ε-vecteur V est associé à une distribution de vérité si pour tout $p \in \{1,\ldots,k\}$, $V_p = V_{p+k}$. Ainsi, avec les ε-vecteurs Z^1, ..., Z^k définis comme suit :

$$\begin{array}{lllllllll} \forall j \in \{1,\ldots,k\}, \forall p \in \{1,\ldots,k\} & \text{si } p = j, & Z_p^j & = & +1 & \text{et} & Z_{p+k}^j & = & -1, \\ & \text{si } p \neq j, & Z_p^j & = & -1 & \text{et} & Z_{p+k}^j & = & +1, \end{array}$$

le ε-vecteur V correspond à une distribution de vérité si et seulement si

$$(\forall j \in \{1,\ldots,k\})(Z^j \cdot V = 0).$$

On remarque de plus que

$$\begin{array}{lcl} C^i \text{ satisfaite sous } D & \Longleftrightarrow & X^i \cdot V \in \{-2, +2, +6\}, \\ C^i \text{ non satisfaite sous } D & \Longleftrightarrow & X^i \cdot V = -6. \end{array}$$

Donc, la clause C^i est satisfaite sous D si et seulement si $X^i \cdot V \geq -2$ avec V correspondant à la distribution de vérité D.

Nous avons désormais la correspondance : la distribution D associée au vecteur V satisfait toutes les clauses C^i si est seulement si les deux propriétés suivantes sont satisfaites :

(a) $\forall i \in \{1,\ldots,q\}, X^i \cdot V \geq -2$.

(b) $\forall j \in \{1,\ldots,k\}, Z^j \cdot V = 0$.

Pour rendre les inégalités (a) relatives à 0, on ajoute deux composantes à chaque vecteur. Ces deux composantes sont fixées à $+1$ pour les vecteurs X^i pour former les vecteurs $\tilde{X}^i$. On force les deux nouvelles composantes de V à $+1$ à l'aide de la transformation des Z^j en $\tilde{Z}^j$ et $\tilde{Z}'$:

$$\begin{array}{rcll} \tilde{X}^i & = & (X^i,+1,+1), & \text{pour tout } i \in \{1,\ldots,q\}, \\ \tilde{Z}^j & = & (Z^j,+1,-1), & \text{pour tout } j \in \{1,\ldots,k\}, \\ \tilde{Z'}^j & = & (Z^j,-1,+1), & \text{pour tout } j \in \{1,\ldots,k\}. \end{array}$$

Ainsi, tous les produits scalaires seront augmentés de 2. Une solution est alors un ε-vecteur $\tilde{V}$ de longueur $2k+2$.

On veut, de plus, transformer les égalités (b) à 0 en inégalités. On définit pour cela les vecteurs

$$\begin{array}{rcll} \tilde{Y}^j & = & -\tilde{Z}^j, & \text{pour tout } j \in \{1,\ldots,k\}, \\ \tilde{Y'}^j & = & -\tilde{Z'}^j, & \text{pour tout } j \in \{1,\ldots,k\}. \end{array}$$

Par conséquent, si I est un sous-ensemble de $\{1,\ldots,q\}$,

$$\left(\begin{array}{l} \\ \forall i \in I, \\ \\ \end{array}\begin{array}{l} C^i \text{ satisfaite sous } D, \\ \text{distribution de vérité} \\ \text{associée à } \tilde{V} \end{array}\right) \Longleftrightarrow \left\{\begin{array}{lcll} \tilde{X}^i \cdot \tilde{V} & \geq & 0 & (\forall i \in I), \\ \tilde{Y}^j \cdot \tilde{V} & \geq & 0 & (\forall j \in \{1,\ldots,k\}), \\ \tilde{Y'}^j \cdot \tilde{V} & \geq & 0 & (\forall j \in \{1,\ldots,k\}), \\ \tilde{Z}^j \cdot \tilde{V} & \geq & 0 & (\forall j \in \{1,\ldots,k\}), \\ \tilde{Z'}^j \cdot \tilde{V} & \geq & 0 & (\forall j \in \{1,\ldots,k\}). \end{array}\right.$$

Considérons la matrice A dont les lignes sont les ε-vecteurs $\tilde{X}^i$ pour $i \in \{1,\ldots,q\}$, $\tilde{Y}^j$, $\tilde{Y'}^j$, $\tilde{Z}^j$ et $\tilde{Z'}^j$ pour $j \in \{1,\ldots,k\}$.

Alors

$$\left(\begin{array}{l} \\ \forall i \in \{1,\ldots,q\}, \\ \\ \end{array}\begin{array}{l} C^i \text{ satisfaite sous } D, \\ \text{distribution de vérité} \\ \text{associée à } \tilde{V} \end{array}\right) \Longleftrightarrow AV \geq 0.$$

On a donc transformé, de façon polynomiale, le problème 3-SAT à q clauses sur k variables en un problème des perceptrons de taille $m \times n$ avec $m = q+4k$ et $n = 2k+2$.

□

4.1.3 Approximation

4.1.3.1 Problème d'optimisation

Nous avons montré que ce problème des perceptrons était difficile à résoudre, mais peut-être un algorithme d'approximation nous permettrait-il de nous rapprocher suffi-

samment d'une solution. Nous allons voir maintenant que ce problème est de plus difficile à approximer. Le problème d'optimisation associé est le suivant :

Problème 40 (Problème d'optimisation* – Max-PP*).
Donnée : *Une ε-matrice A de taille $m \times n$.*
Question : *Trouver un ε-vecteur Y qui maximise le nombre de composantes positives dans le vecteur $A \cdot Y$.*

4.1.3.2 Propriétés

Théorème 41 .
Ce problème d'optimisation Max-PP *est* Max-$\mathcal{SNP}$*-dur.*

Preuve. Réduisons une instance $\mathcal{C}$ de Max-2-Sat en une instance A de Max-PP selon une L-réduction. Soit donc $\mathcal{C} = \{C^1, \ldots, C^q\}$ une instance de Max-2-Sat. Chaque clause C^i est une disjonction de deux littéraux sur les variables x_1, ..., x_k. Nous la noterons $C^i = \{y^i, z^i\}$ avec $y^i \in \{x_{m_i}, \bar{x}_{m_i}\}$ et $z^i \in \{x_{n_i}, \bar{x}_{n_i}\}$. À chaque clause C^i, on associe les ε-vecteurs de taille $2k$:

$$
\begin{array}{lllllllll}
X^i & : & X^i_p & = & +1 & \text{et} & X^i_{p+k} & = & -1 \quad \text{pour } 1 \le p \le k, p \neq m_i, n_i \\
 & & X^i_p & = & +1 & \text{et} & X^i_{p+k} & = & +1 \quad \text{si } x_p \in C_i \\
 & & X^i_p & = & -1 & \text{et} & X^i_{p+k} & = & -1 \quad \text{si } \bar{x}_p \in C_i \\
Y^i & : & Y^i_p & = & +1 & \text{et} & Y^i_{p+k} & = & -1 \quad \text{pour } 1 \le p \le k, p \neq m_i \\
 & & Y^i_{m_i} & = & -1 & \text{et} & Y^i_{m_i+k} & = & +1 \\
Z^i & : & Z^i_p & = & +1 & \text{et} & Z^i_{p+k} & = & -1 \quad \text{pour } 1 \le p \le k, p \neq n_i \\
 & & Z^i_{n_i} & = & -1 & \text{et} & Y^i_{n_i+k} & = & +1 \\
W^i & : & W^i_p & = & +1 & \text{et} & W^i_{p+k} & = & -1.
\end{array}
$$

La redondance des W^i est nécessaire pour la suite. Puis, on définit

$$
\begin{aligned}
\tilde{Y}^i &= -Y^i, \\
\tilde{Z}^i &= -Z^i, \\
\tilde{W}^i &= -W^i.
\end{aligned}
$$

Considérons la matrice A dont les lignes sont les ε-vecteurs X^i, Y^i, $\tilde{Y}^i$, Z^i, $\tilde{Z}^i$, W^i et $\tilde{W}^i$ pour $i = 1, \ldots, q$. A est alors une instance de Max-PP. Le but est de déterminer un ε-vecteur V dont le nombre de composantes positives de son produit avec A est maximal.

Soit une distribution de vérité D de l'instance de 2-Sat initiale $\mathcal{C}$ qui optimise le nombre s de clauses satisfaites. Alors en posant $V_i = V_{i+k} = +1$ si $x_i \in D$, ou $V_i =$

$V_{i+k} = -1$ sinon, V rend $6q+s$ composantes du produit AV positives. Montrons que cette réduction est linéaire :

1. Tout d'abord, on sait qu'on peut rendre au moins une clause sur deux vraie, ce qui signifie que $Opt(\mathcal{C}) \geq q/2$. En revanche, la matrice A est de taille $2k \times 7q$, donc $Opt(A) \leq 7 \cdot q$. Donc
$$Opt(A) \leq 14 \cdot Opt(\mathcal{C}).$$

2. Ensuite, soient U, un ε-vecteur de coût $c'(U) = |\{i|(AU)_i \geq 0\}|$ et Δ, la distribution définie par :
$$\begin{cases} x_i \in \Delta & \text{si } U_i = +1, \\ \bar{x}_i \in \Delta & \text{si } U_i = -1. \end{cases}$$

   Cas 1 : $c'(U) \leq 6q$, $|Opt(A) - c'(U)| \geq 6q + s - c'(U) \geq s \geq |Opt(\mathcal{C}) - c(D)|$ où $c(D)$ est le coût de n'importe quelle distribution de vérité D de l'instance $\mathcal{C}$. Le coût d'une telle distribution D est le nombre de clauses de $\mathcal{C}$ satisfaites sous D. En particulier, $|Opt(A) - c'(U)| \geq |Opt(\mathcal{C}) - c(\Delta)|$.

   Cas 2 : $c'(U) = 6q + t$ où $t > 0$, alors posons $E = \{i|U_i = -U_{i+n}\}$. Définissons le vecteur suivant : $V_{i+k} = V_i = U_i$, pour tout $i \in E$ et $V_i = U_i$ pour $i = \varepsilon k + j$ où $\varepsilon \in \{0, +1\}$ et $j \notin E$.

   Ainsi, pour tout i :

   - si $m_i \in E$ ou $n_i \in E$, alors au moins une des six inégalités ($W^i \cdot U \geq 0$, $\tilde{W}^i \cdot U \geq 0$, $Y^i \cdot U \geq 0$, $\tilde{Y}^i \cdot U \geq 0$, $Z^i \cdot U \geq 0$ et $\tilde{Z}^i \cdot U \geq 0$) n'est pas vérifiée avec U. Avec V, elles le sont toutes. On a donc gagné au moins une inégalité. En revanche, on peut ne plus avoir $X_i \cdot V \geq 0$: au plus une inégalité de perdue. Soit, au pire, autant d'inégalités d'indice i précédemment vérifiées sont vérifiées avec V.
   - si $m_i, n_i \notin E$, alors les inégalités $W^i U \geq 0$, $\tilde{W}^i U \geq 0$, $Y^i U \geq 0$, $\tilde{Y}^i U \geq 0$, $Z^i U \geq 0$ et $\tilde{Z}^i U \geq 0$ étaient vérifiées et le restent avec V.
     Quant à l'inégalité $X^i \cdot U \geq 0$, elle garde son état avec V.

   Alors au moins $6q+t$ inégalités sont satisfaites avec ce nouveau vecteur V. Sachant que pour tout i,
$$\begin{array}{lll} Y^i \cdot V \geq 0, & Z^i \cdot V \geq 0, & W^i \cdot V \geq 0, \\ \tilde{Y}^i \cdot V \geq 0, & \tilde{Z}^i \cdot V \geq 0, & \tilde{W}^i \cdot V \geq 0, \end{array}$$
   la distribution Δ, définie ci-dessus, satisfait au moins t clauses. Donc $c(\Delta) \geq t$.
$$|Opt(A) - c'(U)| = 6q + s - 6q - t = s - t \geq |Opt(\mathcal{C}) - c(\Delta)|.$$

   Dans tous les cas, la distribution de vérité Δ, construite en temps polynomial, vérifie :
$$|Opt(A) - c'(U)| \geq |Opt(\mathcal{C}) - c(\Delta)|.$$

□

Théorème 42 .
Ce problème d'optimisation est approximable à 2 près.

Preuve. En effet, choisissons un ε-vecteur, Z, de façon aléatoire.
Posons $\nu = |\{i|(AZ)_i \geq 0\}|$.

- si $\nu \geq m/2$, on pose $Y = Z$
- sinon, $Y = -Z$

Alors, Y approxime l'optimum à moins d'un facteur 2 près. □

4.2 Problème des Perceptrons Permutés

Le problème des Perceptrons étant dans $\mathcal{NP}$, il admet une preuve interactive a divulgation nulle de connaissance [47]. Étant de plus difficile à résoudre, il permettrait la conception d'un protocole d'identification interactif à divulgation nulle de connaissance. Cependant, afin d'obtenir un protocole plus simple et plus sûr, nous allons définir une variante de ce problème :

Problème 43 (Problème des Perceptrons Permutés – PPP).
Donnée : *Une ε-matrice A de taille $m \times n$ et un multi-ensemble S de m entiers positifs.*
Question : *Existe-t-il un ε-vecteur Y de taille n tel que $\{\{(AY)_i|i = 1, \ldots m\}\} = S$?*

Théorème 44 .
Ce problème des perceptrons permutés ***PPP*** *est $\mathcal{NP}$-complet.*

Preuve. Pour cette preuve, nous allons réduire le problème One-In-Three 3-Sat [43] à notre problème **PPP**.

Problème 45 (*One-In-Three 3-Sat*).
Donnée : *Un ensemble de variables, U, et une liste de 3-clauses, C, sur les variables de U.*
Question : *Existe-t-il une distribution de vérité pour les variables de U telle que toute clause c de C ait un et un seul littéral de vrai ?*

Lemme 46 (Garey et Johnson [43]).
Le problème One-In-Three 3-Sat *est $\mathcal{NP}$-complet.*

En effectuant les mêmes transformations que lors de la réduction de 3-SAT à **PP**, on obtient une réduction de ONE-IN-THREE 3-SAT à **PPP** :

$$\left(\begin{array}{c}\forall i \in \{1,\ldots,q\},\\ C^i \text{ a exactement un littéral vrai sous } D,\\ \text{distribution de vérité associée à } \tilde{V}\end{array}\right) \Longleftrightarrow \left\{\begin{array}{rcll}\tilde{X}^i \cdot \tilde{V} & = & 0 & (\forall i \in \{1,\ldots,q\})\\ \tilde{Z}^j \cdot \tilde{V} & = & 0 & (\forall j \in \{1,\ldots,k\})\\ \tilde{Z'}^j \cdot \tilde{V} & = & 0 & (\forall j \in \{1,\ldots,k\})\end{array}\right.$$

Considérons la matrice A dont les lignes sont les ε-vecteurs $\tilde{X}^i$ pour $i \in \{1,\ldots,q\}$, $\tilde{Z}^j$ et $\tilde{Z'}^j$ pour $j \in \{1,\ldots,k\}$, ainsi que le multi-ensemble S composé de $2k+q$ fois l'élément nul. On a ainsi associé une instance de **PPP** de taille $m \times n$ où $n = 2k$ et $m = q + 2k$ à une instance de ONE-IN-THREE 3-SAT.

Ceci montre que même le sous-problème de **PPP** où tous les éléments de S sont identiques (tous nuls) est $\mathcal{NP}$-complet. □

4.3 Dimensions nécessaires

Comme nous venons de le voir, **PPP** est un problème $\mathcal{NP}$-complet, donc difficile à résoudre dans le cas le pire (algorithmes uniquement exponentiels *a priori*). Mais peut-être faut-il prendre de très grandes tailles pour obtenir des instances, en moyenne, difficiles. Dans ce cas, le problème n'aurait aucun intérêt cryptographique. Un protocole nécessiterait des clés volumineuses ainsi que des communications coûteuses. Nous allons donc expérimenter un certain nombre d'attaques contre ce problème afin d'en évaluer la difficulté pratique. Ceci nous permettra de proposer des tailles sûres.

Ainsi que nous allons le voir par la suite, les valeurs de m et de n dépendront de l'efficacité des attaques. Cependant, on peut déjà chercher une relation entre m et n qui optimisera la difficulté à n fixé. En effet, si m est trop grand, le problème sera trop contraint. En revanche, si m est trop petit, de nombreux ε-vecteurs seront solutions. Dans les deux cas, l'attaque en sera accélérée. Il faut donc prendre m et n tels que le problème admette environ une solution en moyenne. Pour cela, il nous faut savoir :

- le nombre de solutions pour **PP**.
- la probabilité d'obtenir un multi-ensemble donné pour S.

4.3.1 Nombre de solutions pour PP

Lorsque l'on connaît l'existence d'au moins une solution, on peut évaluer, par une méthode combinatoire, le nombre total de solutions pour **PP**. Soient $\alpha \in \{-n,\ldots,n\}$ puis X et V deux ε-vecteurs tels que $X \cdot V = \alpha$. On peut aisément déduire le nombre de composantes identiques :

$$\left\{\begin{array}{rcl}\#\{X_i = V_i\} & = & \dfrac{n+\alpha}{2}\\ \#\{X_i \neq V_i\} & = & \dfrac{n-\alpha}{2}\end{array}\right.$$

Pour tous entiers k, β, γ, δ et pour tout ε-vecteur W tels que

$$\begin{cases} d_H(W,V) &= k \\ X \cdot W &= \delta \end{cases} \text{ et } \begin{cases} \#\{W_i \neq V_i \text{ et } X_i = V_i\} &= \beta \\ \#\{W_i \neq V_i \text{ et } X_i \neq V_i\} &= \gamma \end{cases}$$

on a,

$$\begin{cases} \beta + \gamma &= k, \\ \dfrac{n+\alpha}{2} - \beta + \gamma &= \dfrac{n+\delta}{2} \end{cases} \qquad \text{(voir figure 4.1).}$$

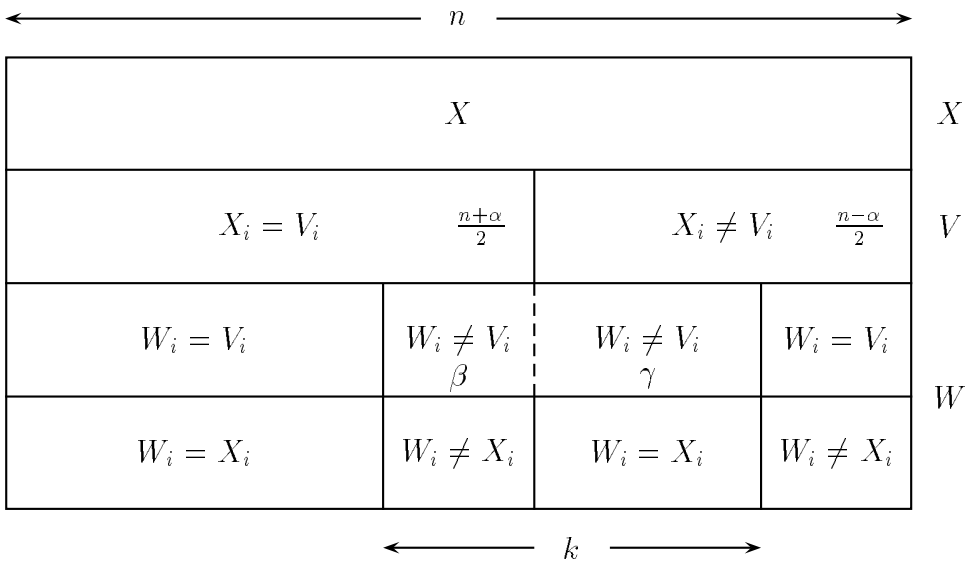

Fig. 4.1 – *Démonstration du nombre de solutions*

Alors,

$$\text{et} \quad \begin{aligned} \beta &= \frac{k}{2} - \frac{\delta-\alpha}{4} \\ \gamma &= \frac{k}{2} + \frac{\delta-\alpha}{4} \end{aligned} \qquad \text{avec} \quad \begin{cases} 0 \leq \beta, \gamma \leq k, \\ 0 \leq \beta \leq \dfrac{n+\alpha}{2}, \\ 0 \leq \gamma \leq \dfrac{n-\alpha}{2}. \end{cases}$$

Par suite, on a une contrainte sur δ :

$$-2 \cdot \min\{k, n+\alpha-k\} \leq \delta - \alpha \leq 2 \cdot \min\{k, n-\alpha-k\}.$$

Considérons les ε-vecteurs W à distance paire de V (*i.e.* $k \in 2\mathbb{N}$):

$$\Pr_{\substack{W \\ d_H(V,W)=k}} [X \cdot W = \delta | X \cdot V = \alpha] = \frac{\binom{\frac{n+\alpha}{2}}{\beta}\binom{\frac{n-\alpha}{2}}{\gamma}}{\binom{n}{k}}.$$

En sommant sur tous les α positifs, on obtient

$$\Pr_{\substack{W \\ d_H(V,W)=k}} [X \cdot W \geq 0 | X \cdot V = \alpha] = \sum_{\delta \in E(n,k,\alpha)} \frac{\binom{\frac{n+\alpha}{2}}{\beta}\binom{\frac{n-\alpha}{2}}{\gamma}}{\binom{n}{k}}$$

en définissant

$$E(n,k,\alpha) = \{\delta \mid \delta \geq 0, -\min\{k, n+\alpha-k\} \leq \frac{\delta-\alpha}{2} \leq \min\{k, n-\alpha-k\} \text{ et } \delta-\alpha \in 4\mathbb{Z}\}.$$

Ainsi, pour tout vecteur Y, le nombre de solutions à distance k, paire, de V, tel que $AV = Y$, est égal à

$$\begin{aligned} N(m,n,k,Y) &= \binom{n}{k} \Pr_{\substack{W \\ d_H(V,W)=k}} [AW \geq 0] \\ &= \binom{n}{k}^{-(m-1)} \prod_{i=1}^{m} \sum_{\delta \in E(n,k,Y_i)} \binom{\frac{n+Y_i}{2}}{\frac{2k-\delta+Y_i}{4}} \binom{\frac{n-Y_i}{2}}{\frac{2k+\delta-Y_i}{4}}. \end{aligned}$$

Et alors, en sommant sur toutes les distances paires possibles k, on obtient le nombre de solutions à distance paire de V, satisfaisant $AV = Y$:

$$N(m,n,Y) = \sum_{\substack{0 \leq k \leq n \\ 2|k}} \binom{n}{k}^{-(m-1)} \prod_{i=1}^{m} \sum_{\delta \in E(n,k,Y_i)} \binom{\frac{n+Y_i}{2}}{\frac{2k-\delta+Y_i}{4}} \binom{\frac{n-Y_i}{2}}{\frac{2k+\delta-Y_i}{4}}.$$

Par conséquent, on peut évaluer le nombre moyen $\tilde{N}(m,n)$ de solutions à distance paire de V, pour une instance de taille $m \times n$. En effet, si on suppose que le vecteur $Y = AV$ suit une distribution Gaussienne, c'est-à-dire

$$\Pr_{\alpha}[Y_i = \alpha] = 2^{-n+1} \binom{n}{\frac{n+\alpha}{2}} \simeq \sqrt{\frac{8}{\pi n}} e^{-\frac{\alpha^2}{2n}},$$

alors,

$$\tilde{N}(m,n) \approx 2 \times \sum_{\substack{0 \leq k \leq n \\ 2|k}} \binom{n}{k}^{-(m-1)} \prod_{\alpha=0}^{\frac{n}{2}-1} \left[\sum_{\delta \in E(n,k,2\alpha+1)} \binom{\frac{n+2\alpha+1}{2}}{\frac{2k-\delta+2\alpha+1}{4}} \binom{\frac{n-2\alpha-1}{2}}{\frac{2k+\delta-2\alpha-1}{4}} \right]^{2m \frac{\binom{n}{\frac{n+2\alpha+1}{2}}}{2^n}}.$$

4.3.2 Probabilité pour chaque multi-ensemble

Soit S^m un multi-ensemble à m éléments.

Notation 47. On notera P_{m,n,S^m} la probabilité pour un vecteur Y qui vérifie $AY \geq 0$ de satisfaire $\{\{(AY)_i\}\} = S^m$.

Notation 48. On notera $|S^m|_j$ le nombre d'éléments de S^m égaux à j.

Alors, si A a un rang maximal,

$$P_{m,n,S^m} = \frac{m!}{|S^m|_1! \ldots |S^m|_n!} \prod_{j=1}^{j=n} p_{n,j}^{|S^m|_j} = m! \prod_{j=1}^{j=n} \frac{p_{n,j}^{|S^m|_j}}{|S^m|_j!}$$

où

$$p_{n,j} = \Pr_{X,Y}[|X \cdot Y| = j] \simeq \sqrt{\frac{8}{\pi n}} e^{-\frac{j^2}{2n}} \text{ si } j \text{ est impair, et 0 sinon.}$$

Il semble clair, et l'expérience le confirme, que la probabilité P_{m,n,S^m} est maximale si S^m a une répartition Gaussienne Σ^m (*i.e.* $|\Sigma^m|_i = mp_{n,i}$). Ainsi, pour toute répartition S^m,

$$P_{m,n,S^m} \leq P_{m,n,\Sigma^m} = m! \prod_{j=1}^{n} \frac{p_{n,j}^{|\Sigma^m|_j}}{|\Sigma^m|_j!} = m! \prod_{j=1}^{n} \frac{p_{n,j}^{mp_{n,j}}}{(mp_{n,j})!}.$$

4.3.3 Conclusion

Avec ces évaluations, on peut dire que le nombre de solutions, suivant un multi-ensemble donné S^m est à peu près

$$\tilde{N}(m,n) \times P_{m,n,S^m} \leq \tilde{N}(m,n) \times P_{m,n,\Sigma^m}.$$

Donc, si on ne veut qu'une solution, il suffit de choisir m et n tels que

$$\tilde{N}(m,n) \times P_{m,n,\Sigma^m} \approx 1.$$

Cette formule ne semble pas admettre d'approximation simple, un certain nombre de ces produits sont alors regroupés dans la figure 4.2.

m	n optimal	Nombre de solutions pour une instance moyenne de **PP**	Probabilité pour Σ^m	Nombre de solutions pour une instance moyenne de **PPP**
71	87	1.3×10^{8}	1.1×10^{-8}	1.3
81	97	5.1×10^{8}	2.0×10^{-9}	1.0
91	107	3.2×10^{9}	3.9×10^{-10}	1.3
101	117	9.4×10^{9}	8.3×10^{-11}	0.8
111	127	4.0×10^{10}	1.8×10^{-11}	0.7
121	137	1.7×10^{11}	8.6×10^{-12}	1.5
131	149	$1,4 \times 10^{12}$	8.9×10^{-13}	1.3
141	157	2.7×10^{12}	2.6×10^{-13}	0.7
151	169	2.1×10^{13}	5.9×10^{-14}	1.2
161	177	5.3×10^{13}	1.8×10^{-14}	1.0
171	187	1.7×10^{14}	5.0×10^{-15}	0.9
181	199	8.0×10^{14}	1.2×10^{-15}	1.0
191	207	2.3×10^{15}	4.1×10^{-16}	1.2
201	217	8.7×10^{15}	1.2×10^{-16}	1.1

Fig. 4.2 – *Dimensions optimales pour* ***PPP***

En particulier, on remarque que les bonnes tailles, dans l'intervalle des dimensions susceptibles d'être intéressantes, sont de la forme $n \approx m + 16$.

4.4 Attaques

Diverses attaques ont été tentées pour mettre à l'épreuve la sécurité de ce problème. Pour ce qui est de **PPP**, en raison de la connaissance du vecteur produit à une permutation près uniquement, peu de manipulations de la matrice seront intéressantes. En effet, lors d'une élimination gaussienne (comme utilisé contre PKP [102], CLE [113] ou tout problème sur les codes correcteurs d'erreurs), il faut appliquer les mêmes transformations au vecteur produit. Ainsi, il semblerait que les recherches exhaustives (plus ou moins subtiles) ou les attaques probabilistes soient les plus performantes.

4.4.1 Vecteur majorité

Le premier vecteur approché qui vient à l'esprit est le *vecteur majorité* M défini par :

$$\text{pour tout } j, \begin{cases} M_j = +1 & \text{si } |\{i | A_{i,j} = +1\}| > \frac{m}{2}, \\ M_j = -1 & \text{sinon.} \end{cases}$$

On va étudier le cas où m et n sont impairs. On note alors p et q les entiers tels que $m = 2q+1$ et $n = 2p+1$. Nous pouvons déjà énoncer un résultat sur le vecteur majorité.

Lemme 49 .
Pour une instance fabriquée, de solution V, la distance de Hamming entre V et M est, en moyenne, de l'ordre de $n \cdot \left(\frac{1}{2} - \frac{1}{\pi}\sqrt{\frac{m}{n}}\right)$.

Preuve. Il nous faut d'abord une loi de distribution pour A et V. Pour cela, nous allons donner une méthode de construction de telles instances aléatoires. On étudiera la distance moyenne de M à V selon cette distribution.

Pour construire une instance (A, V), on choisit le ε-vecteur V aléatoirement. On choisit ensuite une ε-matrice A aléatoire. Pour chaque ligne A_i : si $A_i \cdot V < 0$, on remplace A_i par $-A_i$.

Soit alors V un ε-vecteur aléatoire. Pour toute ligne aléatoire A_i, la probabilité que r éléments correspondent avec V est égale à $2^{-n}\binom{n}{r}$. Donc après remplacement éventuel de A_i par $-A_i$, cette probabilité est $\Pr[A_i \cap V = r] = 2^{-(n-1)}\binom{n}{r}$. L'espérance du nombre de composantes identiques est donc égale à $2^{-(n-1)} \sum_{2r>n} r\binom{n}{r}$.
Soit j l'indice d'une colonne. On peut supposer que la probabilité pour que A_i et V coïncident en cet indice j est égale à

$$F_n = \frac{1}{n} 2^{-(n-1)} \sum_{2.r>n} r\binom{n}{r} = \frac{1}{2} + 2^{-n}\binom{2p}{p} \simeq \frac{1}{2} + \frac{1}{\sqrt{2\pi n}}.$$

Alors, la probabilité pour que plus de la moitié des A_i coïncident avec V en l'indice j est égale à

$$G_{m,n} = \sum_{2s>m} \binom{m}{s} (1 - F_n)^{m-s} {F_n}^s$$

$$\simeq \frac{1}{2^m}\sum_{s=q+1}^{m}\binom{m}{s}+\frac{1}{2^m}\sum_{s=0}^{q}\binom{m}{s}\left[(1+\frac{1}{\sqrt{\pi p}})^{m-s}(1-\frac{1}{\sqrt{\pi p}})^{s}-1\right]$$

$$\simeq \frac{1}{2}+\frac{1}{2^m}\sum_{s=0}^{q}\binom{m}{s}\frac{m-2s}{\sqrt{\pi p}}\simeq\frac{1}{2}+\frac{1}{\pi}\sqrt{\frac{m}{2p}}.$$

D'où

$$G_{m,n}\simeq\frac{1}{2}+\frac{1}{\pi}\sqrt{\frac{m}{n}}.$$

Par conséquent, la distance de Hamming entre V et M est de l'ordre de $\left(\frac{1}{2}-\frac{1}{\pi}\sqrt{\frac{m}{n}}\right)\cdot n$.
□

Une première attaque consiste donc à changer 20 % des composantes du vecteur majorité M, et d'essayer le produit. Mais il y a $\binom{n}{0.20n}$ manières de choisir les composantes de M à changer. Cette première attaque nous oblige à prendre $n \geq 95$, pour amener la recherche à une complexité supérieure à 2^{64}.

Pour affiner cette attaque, on peut choisir en priorité les composantes de M dont les valeurs sont litigieuses (celles pour lesquelles $|\{i|(A_{i,j}=+1\}|$ est proche de $n/2$). Mais de façon surprenante, des composantes de majorité écrasante peuvent être erronées et, en moyenne, 80 % des composantes devront être manipulées. Cette «amélioration» n'impose pas d'augmentation de taille.

4.4.2 Faiblesse

On peut utiliser la faiblesse du problème **PPP** par rapport à **PP**. C'est-à-dire la connaissance du multi-ensemble que forment les composantes du produit. Cependant, il est encore trop coûteux d'essayer toutes les permutations possibles (environ $m!$), mais ce nombre peut être sensiblement réduit en regroupant :

$$\begin{cases} \mathcal{A}_1 &= \{i|\text{un nombre pair de composantes de } A_i \text{ sont à } +1\} \\ \mathcal{A}_2 &= \{i|\text{un nombre impair de composantes de } A_i \text{ sont à } +1\} \\ \mathcal{B}_1 &= \{i|\{(AV)_i = 1 \bmod 4\}\} \\ \mathcal{B}_2 &= \{i|\{(AV)_i = 3 \bmod 4\}\} \end{cases}$$

Comme nous avons pris n impair, on a

$$\begin{cases} \#\mathcal{A}_1 = \#\mathcal{B}_1, & \text{et} & \#\mathcal{A}_2 = \#\mathcal{B}_2 & (1) \\ & \text{ou bien} & & \\ \#\mathcal{A}_1 = \#\mathcal{B}_2, & \text{et} & \#\mathcal{A}_2 = \#\mathcal{B}_1 & (2) \end{cases}$$

En effet, si V contient α éléments à $+1$ et si A_i en contient β_i, alors le produit scalaire $(AV)_i$ est égal à $(n+2\alpha)+2\beta_i$ modulo 4.

Supposons que notre instance satisfasse le (1). Alors il y a beaucoup moins de permutations à essayer. Cependant, la complexité rend toujours les calculs inimaginables $((\frac{m}{2})!^2)$. En revanche, cette approche nous permet de déterminer la parité de $+1$ (ou de -1) du vecteur solution. En effet, en changeant deux par deux les coordonnées de V, le

résidu modulo 4 de $(AV)_i$ reste inchangé. Soit alors A_i une ligne possédant un nombre pair de composantes à $+1$, et multiplions la par le vecteur U dont toutes les composantes sont à -1. Le produit scalaire A_iU est alors égal à n modulo 4. Si $n = 1 \bmod 4$, alors $A_iU = A_iV = 1 \bmod 4$. Ce qui signifie que V a, comme U, un nombre pair de composantes à $+1$.

4.4.3 Le recuit simulé

Vue l'inefficacité des attaques précédentes, nous avons tenté l'algorithme probabiliste bien connu, notamment en intelligence artificielle, du *recuit simulé* [110]. Cet algorithme tente de minimiser, de façon probabiliste, une fonction d'énergie définie sur un espace métrique fini. C'est une amélioration de la méthode du gradient. Tandis que cette dernière se trouve bloquée lorsqu'elle atteint un minimum local, le recuit simulé essaie d'en sortir par des perturbations aléatoires. L'amplitude de ces perturbations est importante au début, puis tend vers zéro.
Comme pour la méthode du gradient, il faut que la fonction d'énergie ait une certaine «continuité», ou régularité, c'est-à-dire qu'elle ne fasse pas des sauts aberrants d'un point à son voisin. En particulier, le recuit simulé semble tout à fait adapté à l'attaque de **PP**, mais ne permettra pas de résoudre directement **PPP**. En d'autres termes, nous allons essayer de résoudre MAX-PP, sachant que l'optimum nous fournira une solution de **PP**.

Algorithme

- On définit tout d'abord la fonction d'énergie que l'on cherchera à minimiser. Dans notre cas, pour résoudre **PP**, on prend la fonction d'énergie suivante que l'on veut annuler : $E(V) = \#\{i | A_i.V < 0\}$.
- Ensuite, on a besoin de définir un voisinage pour tout vecteur, ici

$$\mathit{Vois}(V) = \{X | d_H(V, X) = 1\}.$$

- Enfin, la performance de l'algorithme dépendra des trois paramètres Θ_i, Θ_f (températures initiale et finale) et τ (taux de décroissance de la température).
- L'algorithme est alors le suivant :

1. Choix d'un vecteur S aléatoire dans l'espace des solutions possibles.
2. On pose $\Theta \longleftarrow \Theta_i$ (température initiale).
3. Choix de $V \in_R \mathit{Vois}(S)$
4. $\Delta \longleftarrow E(V) - E(S)$
5. si $\Delta > 0$ alors $p \longleftarrow \exp(-\Delta/\Theta)$, sinon $p \longleftarrow 1$.
6. $S \xleftarrow{\text{avec proba } p} V$
7. $\Theta \longleftarrow \Theta.\tau$
8. si $(E(S) > 0) \wedge (\Theta > \Theta_f)$ retourner en 3

Variantes On peut améliorer cette attaque en utilisant la remarque précédente :

- On prend le vecteur initial S aléatoire parmi les vecteurs ayant la bonne parité de composantes à $+1$.
- On définit le voisinage $Vois(V) = \{X | d_H(X, V) = 2\}$. Ainsi, tous les vecteurs testés auront la bonne parité de $+1$.

Performances Cette méthode optimisée se révèle être la plus efficace. Nous avons effectué de nombreux tests sur des matrices ayant les tailles proposées figure 4.3. En quelques minutes, on peut trouver une solution pour toute instance de **PP** de taille allant jusqu'aux environs de 200.

m	n	temps (s)
71	87	6
81	97	8
91	107	10
101	117	12
111	127	18
121	137	25
131	149	50
141	157	70
171	187	180
201	217	800

Fig. 4.3 – *Temps moyen sur les instances «faciles» **PP***

Fig. 4.4 – *Courbe de densité du temps moyen pour une solution de **PP**(101×117)*

Un exemple de courbe de densité du temps de calcul pour trouver une solution de **PP** de taille 101×117 est présenté sur la figure 4.4.

Les moyennes sont prises en effectuant 50 tentatives pour chaque instance. Quant à cette courbe, elle est construite à partir de 1000 instances, en excluant les temps anormalement grands. On observe le pic aux environs de 11 secondes.

Nous avons fait tourner ces attaques pendant plusieurs moins, jamais une solution pour **PPP**, pour des tailles supérieures à 71 (avec $n \approx m+16$), n'a été trouvée. Notre recherche de solution par recuit simulé trouve une solution quelconque de **PP**. La probabilité pour qu'une solution de **PPP** soit obtenue dépend du nombre total de solutions. Pour trouver la bonne solution avec probabilité p, en supposant qu'on obtient, avec probabilité uniforme, toutes les solutions à distance paire de la bonne solution, il faut réitérer l'attaque :

$$1 - p = \Pr[V \text{ pas trouvé en } k \text{ tours}] = \left(1 - \frac{1}{n}\right)^k \simeq e^{-\frac{k}{n}}.$$

Ainsi,

$$k \simeq n \cdot \ln\left(\frac{1}{1-p}\right).$$

Pour trouver la bonne solution avec probabilité supérieure à p, il faut donc réitérer la recherche $-\ln(1-p) \cdot n$ fois. Par conséquent, après $0.7 \times n$ itérations, on a une chance sur deux de trouver la bonne solution. On peut alors estimer le temps moyen sur notre machine pour avoir une chance sur deux de trouver la bonne solution (voir figure 4.5).

taille	nombre de solutions	temps pour une solution de **PP** (secondes)	temps pour une solution de **PPP** avec $Pr = 1/2$ (secondes)		facteur* de travail
71×87	$1.3\ 10^8$	6	$2.7.10^8$	8 ans	54
81×97	$5.1\ 10^8$	8	$1.4.10^9$	45 ans	56
91×107	$3.2\ 10^9$	10	$1.1.10^{10}$	350 ans	59
101×117	$9.4\ 10^9$	12	$3.9.10^{10}$	1250 ans	61
111×127	$4.0\ 10^{10}$	18	$2.5.10^{11}$	8000 ans	64
121×137	$1.7\ 10^{11}$	25	$1.5.10^{12}$	$47\ 10^3$ ans	66
131×149	$1{,}4\ 10^{12}$	50	$2.5.10^{13}$	$780\ 10^3$ ans	70
141×157	$2.7\ 10^{12}$	70	$6.6.10^{13}$	$2\ 10^6$ ans	72
171×187	$1.7\ 10^{14}$	180	$1.0.10^{16}$	$340\ 10^6$ ans	79
201×217	$8.7\ 10^{15}$	800	$2.4.10^{18}$	$77\ 10^9$ ans	87

* facteur de travail estimé en utilisant le fait que la machine utilisée tourne aux environs de 60 à 70 MIPS

Fig. 4.5 – *Facteur de travail de l'attaque de **PPP** par recuit simulé*

Ces diverses expériences nous permettent de proposer des tailles au problème pour une utilisation sûre en cryptographie. Pour qu'une attaque de ce type nécessite un facteur de travail supérieur à 2^{60}, on pourra utiliser des tailles à partir de $m = 101$ et $n = 117$.

A priori, quelle que soit l'attaque probabiliste, elle ne pourra différencier la solution de **PPP** des nombreuses solutions de **PP**. Alors même en supposant une attaque de l'ordre de la milliseconde contre **PP** (un problème $\mathcal{NP}$-complet) pour une instance de taille 121×137, le temps moyen pour trouver une solution au problème **PPP** reste aux environs de l'année. 121×137 semble donc une taille raisonnable pour se prémunir contre d'éventuelles attaques à venir.

4.5 Application à la cryptographie

En raison des faibles tailles suffisantes pour obtenir un problème réellement difficile, un protocole d'identification se trouve être tout à fait réalisable.

Nous allons en présenter deux versions. La première est à trois passes, mais nécessite un grand nombre d'itérations pour atteindre une bonne sécurité. La variante à cinq passes diminue légèrement ce nombre d'itérations, et limite par la même occasion le coût des transmissions.

4.5.1 Protocole d'identification à 3 passes

La première version à 3 passes est présentée figure 4.6. Lors de l'authentification, le prouveur commence par fabriquer une nouvelle instance dérivée du problème (A, S) public. Pour cela, il permute les lignes de la matrice A à l'aide d'une permutation P, puis il permute les colonnes tout en changeant aléatoirement les signes à l'aide d'une permutation «signée» Q. Il obtient ainsi une nouvelle matrice A'. En faisant opérer Q^{-1} sur V, la clé secrète solution du problème (A, S), il obtient une solution V' du problème (A', S). Il choisit ensuite un vecteur aléatoire W dans $\mathbb{Z}/p\mathbb{Z}$. Ce vecteur servira à masquer V' en $R = W + V' \bmod p$. Enfin, il met en gage (P, Q), W, $A'W$, R et $A'R$ et attend la question du vérifieur. Les questions $c = 0$ et $c = 1$ servent à vérifier l'honnêteté du prouveur quant à la construction de tous ces objets. La question $c = 2$ permet de vérifier que le témoin V utilisé satisfait bien $\{\{(AV)_i\}\} = S$. Quant à la dernière question, elle permet de tester si le témoin est bien un ε-vecteur.

Théorème 50 .
Ce protocole est un système interactif de preuve pour ***PPP****.*

Preuve. Montrons donc qu'il est consistant puis significatif.

- Alice (qui connaît le secret V) saura répondre aux 4 questions, et donc convaincre Bob, le vérifieur, avec probabilité 1.

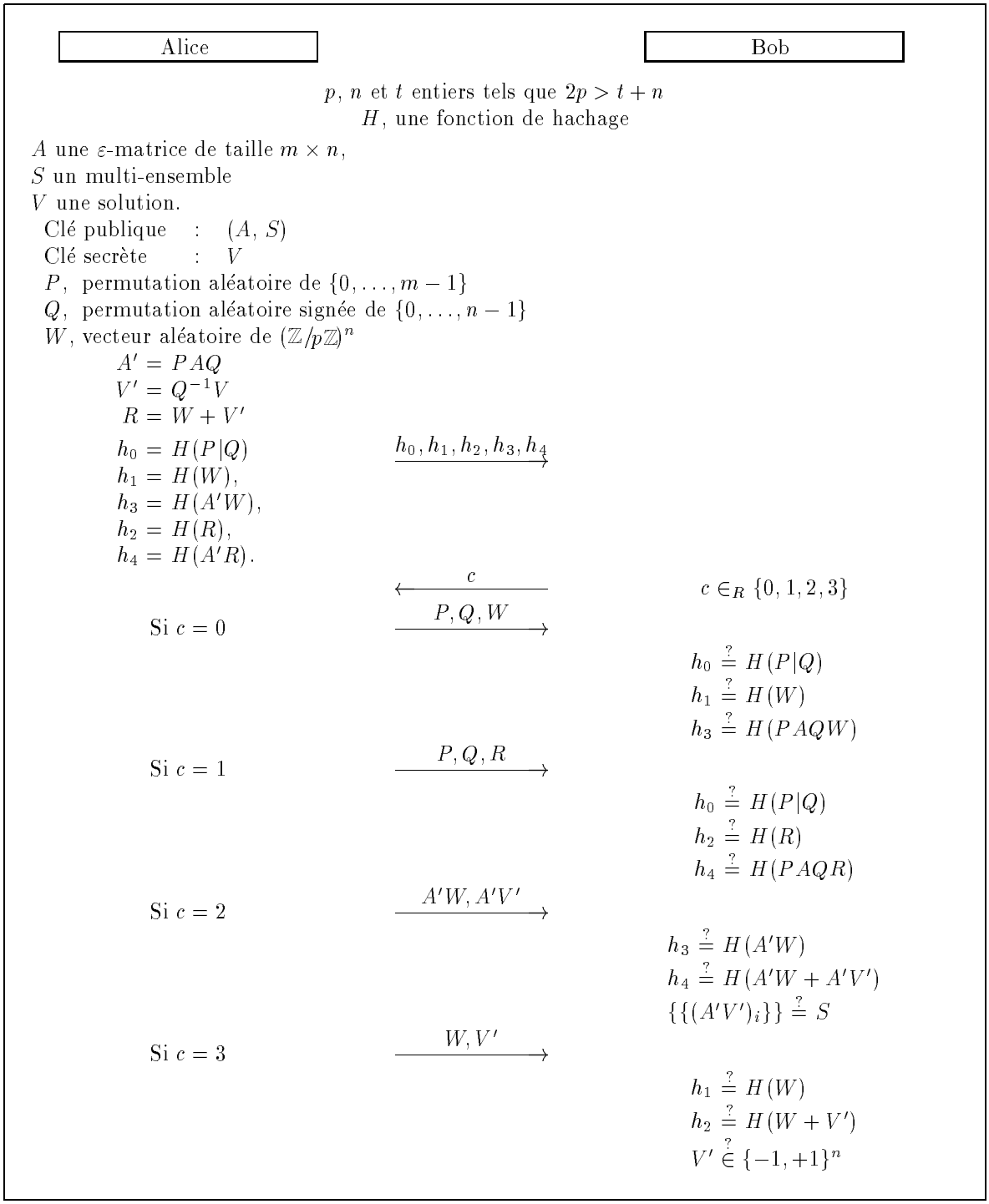

Fig. 4.6 – *Schéma des Perceptrons Permutés à 3 passes*

– Pour toute constante k, on peut énoncer le résultat suivant :

Lemme 51 .
Une fraude avec probabilité supérieure à $\left(\frac{3}{4}\right)^k + \varepsilon$, avec $\varepsilon \geq 1/P(n)$ où n est la taille du problème, après k itérations de ce protocole est équivalente au problème des perceptrons permutés ou à la recherche de collisions de la fonction de mise en gage.

Preuve. Supposons que Charlie soit capable de se faire passer pour Alice avec probabilité supérieure à $\left(\frac{3}{4}\right)^k + \varepsilon$, avec $\varepsilon \geq 1/P(n)$ où n est la taille du problème, après k itérations de ce protocole.

Nous allons alors construire une machine de Turing Polynomiale Probabiliste qui extrait soit une solution au problème des perceptrons permutés, soit une collision pour la fonction de mise en gage, avec une probabilité non négligeable.

Considérons l'arbre d'exécution $T(\omega)$ correspondant à toutes les questions du vérifieur sur k tours quand Charlie a ω pour ruban aléatoire.

$$\alpha = \Pr_{\omega}[T(\omega) \text{ a un sommet avec 4 fils}]$$

Il est clair que $\alpha \geq \varepsilon$. Choisissons alors un ruban aléatoire quelconque : avec probabilité supérieure à $1/P$, $T(\omega)$ possède un sommet avec quatre fils.

Effectuons toutes les identifications possibles (2^k) afin de déterminer les sommets avec quatre fils. Un tel sommet correspond à la situation où les cinq mises en gages h_0, h_1, h_2, h_3 et h_4 ont été effectuées, et Charlie peut répondre aux quatre questions de Bob. Ces réponses vérifient :

$$\begin{array}{cccccccccc} & & & & & H(P_0|Q_0) = h_0 = H(P_1|Q_1), & & & & \\ H(W_0) & = & h_1 & = & H(W_3), & H(R_1) & = & h_2 & = & H(W_3 + V_3'), \\ H(P_0AQ_0W_0) & = & h_3 & = & H(Y_2), & H(P_1AQ_1R_1) & = & h_4 & = & H(Y_2 + Z_2). \end{array}$$

À moins d'avoir trouvé une collision pour H, on peut considérer

$$\begin{array}{ccccc} P = P_0 = P_1, & & & R = R_1 = W_3 + V_3' = W + V', \\ Q = Q_0 = Q_1, & \text{et} & & Y = Y_2 = P_0AQ_0W_0 = PAQW, \\ W = W_0 = W_3, & & & Y + Z = Y_2 + Z_2 = P_1AQ_1R_1 = PAQR \end{array}$$

tels que $V' \in \{-1, +1\}^n$ et $\{\{Z_i\}\} = S$. Alors

$$Y + Z = PAQR = PAQW + Z = PAQW + PAQV',$$

donc $Z = PAQV'$. En posant $V = QV'$, on a $Z = PAV$, d'où $\{\{(AV)_i\}\} = S$. Par conséquent, on a résolu le problème des perceptrons permutés avec probabilité non négligeable. □

$$Pr[\text{Charlie convainque Bob après } k \text{ tours}] \leq \left(\frac{3}{4}\right)^k + o(1/poly(n)).$$

□

Théorème 52 .
Dans le modèle de l'Oracle Aléatoire, ce protocole est un système interactif de preuve à divulgation nulle de connaissance.

Preuve. Tout d'abord, on suppose que $m \leq n$ et que la matrice A est de rang m. On veut montrer que les interactions entre Alice et Bob (et même Charlie) peuvent être simulées par une machine de Turing Polynomiale Probabiliste avec une distribution indistinguable.

Nous allons simuler les interaction avec Charlie, un vérifieur malhonnête. Soit f sa stratégie probabiliste, c'est-à-dire que pour toute mise en gage initiale h_0, h_1, h_2, h_3, h_4, $f(\omega, \text{historique}, h_0, h_1, h_2, h_3, h_4)$ renvoie une question entre 0 et 3

Voici la Machine de Turing Polynomiale Probabiliste $\mathcal{S}$ qui construit un ruban de communication indistinguable d'un ruban créé au cours d'une réelle identification entre Alice et Charlie.

1. $\mathcal{S}$ choisit une question aléatoire $C \in \{0, 1, 2, 3\}$

$C = 0$ $\mathcal{S}$ choisit aléatoirement P, Q et W,
calcule $h_0 = H(P, Q)$, $h_1 = H(W)$, $h_3 = H(PAQW)$
et choisit h_2 et h_4, chaînes aléatoires.
Alors, $x = \text{concat}(h_0, h_1, h_2, h_3, h_4)$ et $y = \text{concat}(P, Q, W)$.

$C = 1$ $\mathcal{S}$ choisit aléatoirement P, Q et R,
calcule $h_0 = H(P, Q)$, $h_2 = H(R)$, $h_4 = H(PAQR)$
et choisit h_1 et h_3, chaînes aléatoires.
Alors, $x = \text{concat}(h_0, h_1, h_2, h_3, h_4)$ et $y = \text{concat}(P, Q, R)$.

$C = 2$ $\mathcal{S}$ choisit un vecteur quelconque Y tel que $\{\{Y_i\}\} = S$, et un vecteur aléatoire X.
Y est supposé être PAV et X, $PAQW$. Il est clair que Y a la même distribution que PAV, mais est-il vrai que la distribution de $PAQW$ est uniforme ?
Soit $Z \in (\mathbb{Z}/p\mathbb{Z})^m$. On suppose PAV fixé, ce qui fixe P. Puisque A est de rang m,

$$\begin{aligned}
\Pr_{Q,W}[PAQW = Z] &= \Pr_{Q,W}[AQW = P^{-1}Z] \\
&= \frac{\#\{(Q,W)|AQW = P^{-1}Z\}}{2^n n! p^n} \\
&= \frac{\sum_Q \#\{W|W = (PAQ)^{-1}Z\}}{2^n n! p^n} \\
&= \frac{\sum_Q p^{n-m}}{2^n n! p^n} = \frac{2^n n! p^{n-m}}{2^n n! p^n} = \frac{1}{p^m}.
\end{aligned}$$

$\mathcal{S}$ calcule $h_3 = H(X)$, $h_4 = H(X+Y)$
et choisit h_0, h_1 et h_2, chaînes aléatoires.
Alors, $x = \text{concat}(h_0, h_1, h_2, h_3, h_4)$ et $y = \text{concat}(X, X+Y)$.

$C = 3$ $\mathcal{S}$ choisit un vecteur aléatoire W, et un ε-vecteur aléatoire E,
calcule $h_1 = H(W)$, $h_2 = H(W+E)$
et choisit h_0, h_3 et h_4, chaînes aléatoires.
Alors, $x = \text{concat}(h_0, h_1, h_2, h_3, h_4)$ et $y = \text{concat}(W, W+E)$.

2. $\mathcal{S}$ évalue $c = f(\omega, \text{historique}, x)$.

3. Si $c = C$ alors $\mathcal{S}$ écrit x, c et y, sinon $\mathcal{S}$ retourne en 1 (reset [48]).

Alors $\mathcal{S}$ simule un ruban de communication indistinguable d'un ruban d'une véritable identification de r tours en un nombre moyen de $4 \times r$ étapes. □

Grâce à ce résultat de complexité, nous pouvons affirmer, comme montré sur le schéma de Fiat-Shamir, que la répétition séquentielle de ce protocole fournit un schéma d'identification sûr contre les attaques actives.

4.5.2 Protocole d'identification à 5 passes

Comme nous l'avons vu, la probabilité de fraude est de 3/4 à chaque tour. Il faut donc réitérer de nombreuses fois pour obtenir une bonne sécurité. Usuellement, on réclame une probabilité de fraude inférieure à un pour un million, ce qui nécessite 48 tours. On va améliorer ceci avec le protocole à 5 passes présenté figure 4.7. Nous avons pour cela appliqué une transformation usuelle sur le protocole à trois passes en regroupant les deux premières questions en une seule grâce à un paramètre k envoyé par le vérifieur.

Théorème 53 .
Ce protocole est un système interactif de preuve pour ***PPP****.*

Preuve. La preuve est analogue est celle présentée pour le schéma à 3 passes. La probabilité de fraude est abaissée aux environs de 2/3. Plus généralement, pour tout k,

Lemme 54 .
Une fraude avec probabilité supérieure à $\left(\frac{2p-1}{3(p-1)}\right)^k + \varepsilon$, *avec* $\varepsilon \geq 1/P(n)$ *où* n *est la taille du problème, après* k *itérations de ce protocole est équivalente au problème des perceptrons permutés ou à la recherche de collisions de la fonction de mise en gage.*

□

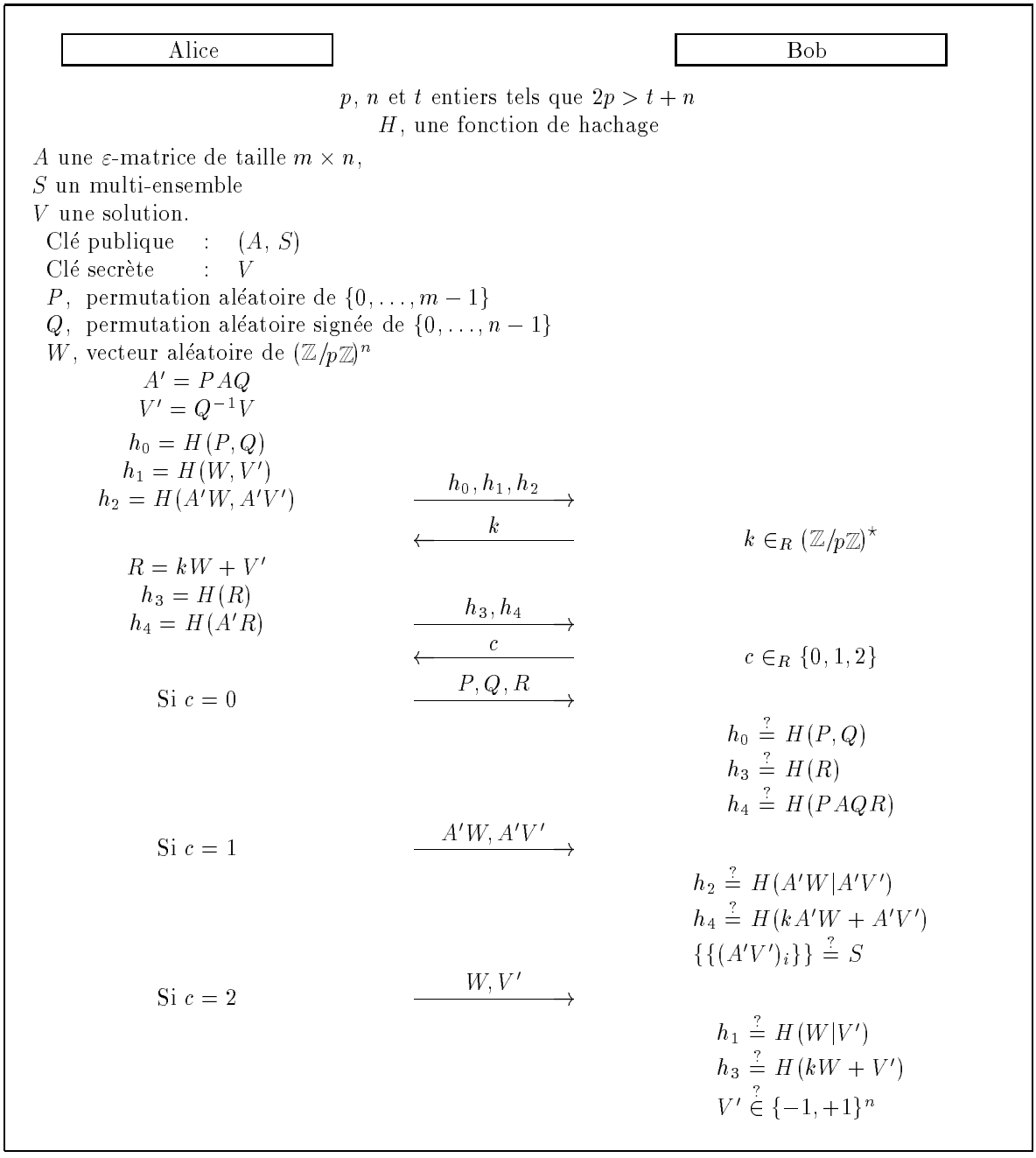

Fig. 4.7 – *Schéma des Perceptrons Permutés à 5 passes*

Théorème 55 .
Dans le modèle de l'Oracle Aléatoire, ce protocole est un système interactif de preuve à divulgation nulle de connaissance.

Preuve. La simulation ressemble à celle présentée pour le schéma à trois passes. L'étape délicate est cette fois ci pour $C = 1$, mais repose également sur la distribution de $PAQW$. Pour ce schéma, le simulateur construit simule un ruban de communication indistinguable d'un ruban d'une véritable identification de r tours en un nombre moyen de $3 \times r$ étapes. □

La répétition séquentielle de ce protocole fournit donc un schéma d'identification sûr contre les attaques actives.

4.5.3 Construction d'une instance

Pour un usage cryptographique, il nous faut pouvoir construire une instance de **PPP** dont on connaisse une solution, sans pour autant restreindre l'ensemble des instances possibles. Il faut donc que toute instance admettant une solution puisse apparaître.

La méthode utilisée pour évaluer la qualité du vecteur majorité convient parfaitement :

- On commence par tirer aléatoirement un ε-vecteur V, qui sera la future solution puis une ε-matrice A aléatoire de taille $m \times n$.
- Pour toute ligne i de la matrice A :
  - Si $(AV)_i < 0$, on remplace la i-ième ligne de A par son opposée.
  - Sinon, on laisse cette ligne inchangée.
- On calcule $S = \{\{(AV)_i | i = 1, \ldots, m\}\}$.

Alors, (A, S) est une instance de **PPP** admettant V pour solution. De plus, par cette construction, toute instance ayant une solution peut apparaître, avec une probabilité proportionnelle au nombre de solutions que l'instance de **PP** associée possède. C'est-à-dire que plus une instance de **PP** a de solutions, plus elle a de chances d'apparaître, ce qui compliquera d'autant l'attaque.

4.5.4 Ensemble fini

Toujours pour une application cryptographique, il faut borner les tailles des nombres manipulés pour les stocker sur un nombre constant de bits. On ramène alors le problème dans un ensemble fini. On considère

- l'anneau à p éléments, $\mathbb{Z}/p\mathbb{Z}$ (dans la suite, p sera pris égal à 251 pour des raisons historiques, donc premier, mais rien ne l'oblige).
- une ε-matrice A de taille $m \times n$, n impair.

– un vecteur T de taille m à composantes entières positives impaires majorées par $t < p$.

On cherche à remplacer le test $AY = T$ par $AY = T \bmod p$, c'est-à-dire qu'un ε-vecteur Y sera accepté si, pour tout i, $(AY)_i \in \{kp + T_i |$ k entier relatif$\}$. Or, pour tout i, la i-ième composante du produit est impaire ainsi que T_i, donc l'ensemble se limite à $\{kp + T_i |\ k$ pair$\}$. Par conséquent, les cas à refuser sont $(AY)_i \leq -2p + T_i$ ou $(AY)_i \geq 2p + T_i$. Or, pour tout i et tout Y, $|(AY)_i| \leq n$, donc la condition $2p > n + t$ entraîne l'équivalence :

$$AY = T \iff AY = T \bmod p.$$

4.5.5 Codage du problème

Nous allons évaluer l'espace mémoire nécessaire au stockage des clés. Pour le minimiser, on peut rendre commun à tout le monde les lignes de la matrice A au signe près. Ainsi, une personne extérieure choisit aléatoirement une ε-matrice M. Cette matrice sera une donnée commune à tous. Puis :

– chaque individu choisit sa clé secrète V dans $\{-1, +1\}^n$.

– sa clé publique sera le couple (L, S) où

  – $L \in \{-1, +1\}^m$ tel que pour tout j, $L_j(MV)_j > 0$
  – $S = \{\{L_j(MV)_j | j = 1, \ldots, m\}\}$

Notation 56. Moyenne(m,n,y) est le nombre moyen d'éléments de S égaux à y pour une instance de taille $m \times n$:

$$\text{Moyenne}(m, n, y) = \lceil \frac{m}{2^{n-1}} \binom{n}{\frac{y+n}{2}} \rfloor.$$

Ainsi, le stockage des clés ne nécessite qu'un espace réduit. Pour $n = 117$ et $m = 101$, on pourra prendre $t = 33$ et $\ell = 3$. On obtient alors les tailles suivantes :

Clé secrète	ε-vecteur V de taille n	n bits	117	117	15 octets
Clé publique	ε-vecteur L de taille m	m bits	101	149	19 octets
	vecteur D de taille $\frac{t-1}{2}$ à composantes dans $\{-2^\ell + 1, \ldots, 2^\ell - 1\}$	$\frac{\ell(t-1)}{2}$ bits	48		

La preuve consistera à montrer la connaissance d'un ε-vecteur X tel que

$$(\forall y \in \{1, 3, \ldots, t\}) \qquad \#\{j | L_j(MX)_j = y\} = D_{\frac{y-1}{2}} + \text{Moyenne}(m, n, y).$$

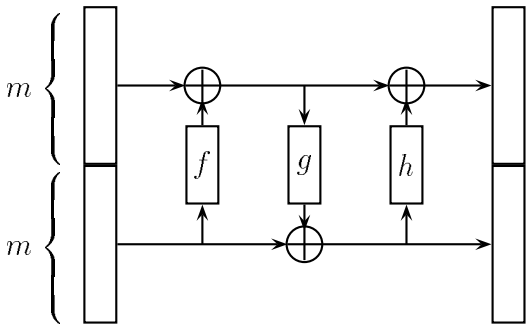

Fig. 4.8 – *Permutations pseudo-aléatoires*

4.6 Astuces pratiques

4.6.1 Les permutations

Fabriquer efficacement des permutations de façon parfaitement aléatoire n'est pas chose aisée, cependant, on peut utiliser les générateurs de permutations basés sur le schéma de Feistel (voir figure 1.2) utilisé pour le DES, étudiés par Luby et Rackoff [64] ainsi que par Patarin [81]. La construction est décrite figure 4.8. Elle permet de fabriquer une permutation aléatoire à partir de trois fonctions aléatoires. Ces trois fonctions aléatoires f, g et h, agissant sur des entrées de petite taille, 3 ou 4 bits, sont fabriquées en tirant leurs tables de valeurs au hasard.

4.6.2 Les engagements

Deux situations peuvent être envisagées selon que l'on préfère optimiser le temps et la quantité des transferts de données, ou si l'on a des contraintes très strictes sur la mémoire vive. Dans le premier cas, on peut effectuer les engagements sous forme d'arbre de hachage. Dans la suite, nous nous efforcerons surtout de minimiser la mémoire nécessaire en raison de la faible capacité de la carte à microprocesseur que nous utiliserons.

4.6.3 Les objets aléatoires

On suppose que l'on possède un générateur de bits pseudo-aléatoires public, de la forme $f(s, i)$ où s est le germe, qui suffit à décrire la séquence des bits suivants, et i le i-ième nombre généré. Un germe de 64 bits semble suffisant. On ne communique, alors, que les germes de chaque objet aléatoire.

4.7 Performances

Il serait maintenant intéressant de comparer ces schémas aux autres schémas d'identification fondés sur des problèmes combinatoires et non sur des problèmes de théorie des nombres, notamment PKP [102], SD [112] et CLE [113]. Les performances sont comparées sur la figure 4.9. Les complexités d'attaque de CLE et PKP sont inspirées de [90].

	SD Stern	CLE Stern	PKP Shamir	**PPP** 3p ZK	**PPP** 5p ZK
Taille de la matrice	256×512	24×24	16×34	101×117	
sur le corps	$\mathbb{F}_2$	$\mathbb{F}_{16}$	$\mathbb{F}_{251}$	$\mathbb{F}_2$	
complexité de la meilleure attaque connue	2^{68}	2^{73}	2^{70}	2^{61}	
Nombre de tours	35	20	20	48	35
clé publique (bits)	256	80	272	149	
clé secrète (bits)	512	80	204	117	
bits envoyés à chaque tour	954	824	741	896	1040
transmission globale (kilo-octets)	4.08	2.01	1.81	5.25	4.44

Fig. 4.9 – *Performances des différents schémas d'identification basés sur des problèmes $\mathcal{NP}$-complets*

- Comme le montre ce tableau, avec une complexité d'attaque comparable aux autres schémas et une probabilité de 1 pour 1 million pour un intrus d'être accepté, une identification nécessitera moins de 4.5 kilo-octets de transmissions entre le prouveur et le vérifieur. Et nous pouvons légèrement diminuer cette quantité en utilisant des arbres de hachage.

- De plus, les opérations se résument à des additions et des soustractions entre petits entiers (moins d'un octet), voire modulo 2. Elles sont parfaitement adaptées à l'environnement minimal d'un processeur 8 bits.

- Les valeurs à stocker sont restreintes. Moins de 15 octets pour la clé secrète et moins de 20 pour la clé publique. Seulement 101 bits de la clé publique sont nécessaires au prouveur. L'espace mémoire nécessaire est alors vraiment très faible, comparé à PKP ou SD.

- Ces protocoles n'utilisent que des opérations très simples, ainsi le programme ne sera pas très encombrant. De plus, la taille des données (données communes et clés) est très faible. Une petite EEPROM sera donc suffisante.

- Peu de calculs intermédiaires doivent être stockés : une RAM minimale suffira.

4.8 Implantation sur carte à microprocesseur

Une implantation sur carte à microprocesseur a été effectuée par Guillaume Poupard [89], et les performances pratiques sont tout à fait encourageantes.

4.8.1 Caractéristiques de la carte

En effet, nous avons implanté la version à trois passes sur une carte très courante à base du microprocesseur 8 bits d'Intel, le 6805. Les caractéristiques de cette carte sont classiques :

- 2 kilo-octets d'EEPROM (mémoire effaçable et réinscriptible) dans lesquels durent être logés le programme, les données communes et les clés. Aucun calcul ne peut être effectué dans cette zone mémoire en raison du temps d'écriture prohibitif.
- 160 octets de RAM. Mais, chose courante dans ce type de carte à faible coût, la pile évolue au milieu de cette RAM, et une bonne trentaine d'octets, non contigus, sont réservés à des fonctions de sécurité ou autres opérations système. La gestion de cette RAM fut donc un véritable casse-tête.
- Le boîtier lecteur de carte est cadencé à 3,57 MHz et permet des débits de communication de 9600 ou 19200 bauds.
- Le protocole de communication $T = 0$ est utilisé.

4.8.2 Structure du programme

La structure du programme de la carte est présentée sur la figure 4.10. Les astuces pratiques précédemment citées ont été utilisées.

- les permutations aléatoires sont fabriquées à partir de trois fonctions aléatoires f, g, et h. La permutation réciproque est alors aisée à calculer.
- les objets aléatoires sont fabriqués à partir d'un germe aléatoire et d'un générateur linéaire congruentiel public.
- pour la fonction de hachage, nous avons recherché une fonction bien adaptée à un processeur 8 bits. Une variante de Snefru [91] retournant un condensé sur 96 bits sembla fournir une sécurité convenable.

4.8.3 Émulateur de carte

Nous avons utilisé un émulateur de carte [36] pour tester le programme en langage machine avant de le faire tourner sur la carte. Cela nous a évité de «planter» irrémédiablement la carte en cas de bug (chose courante en programmation en langage machine). Nous avons de plus pu déterminer précisément le nombre de cycles que la carte avait à effectuer, et ainsi optimiser le protocole.

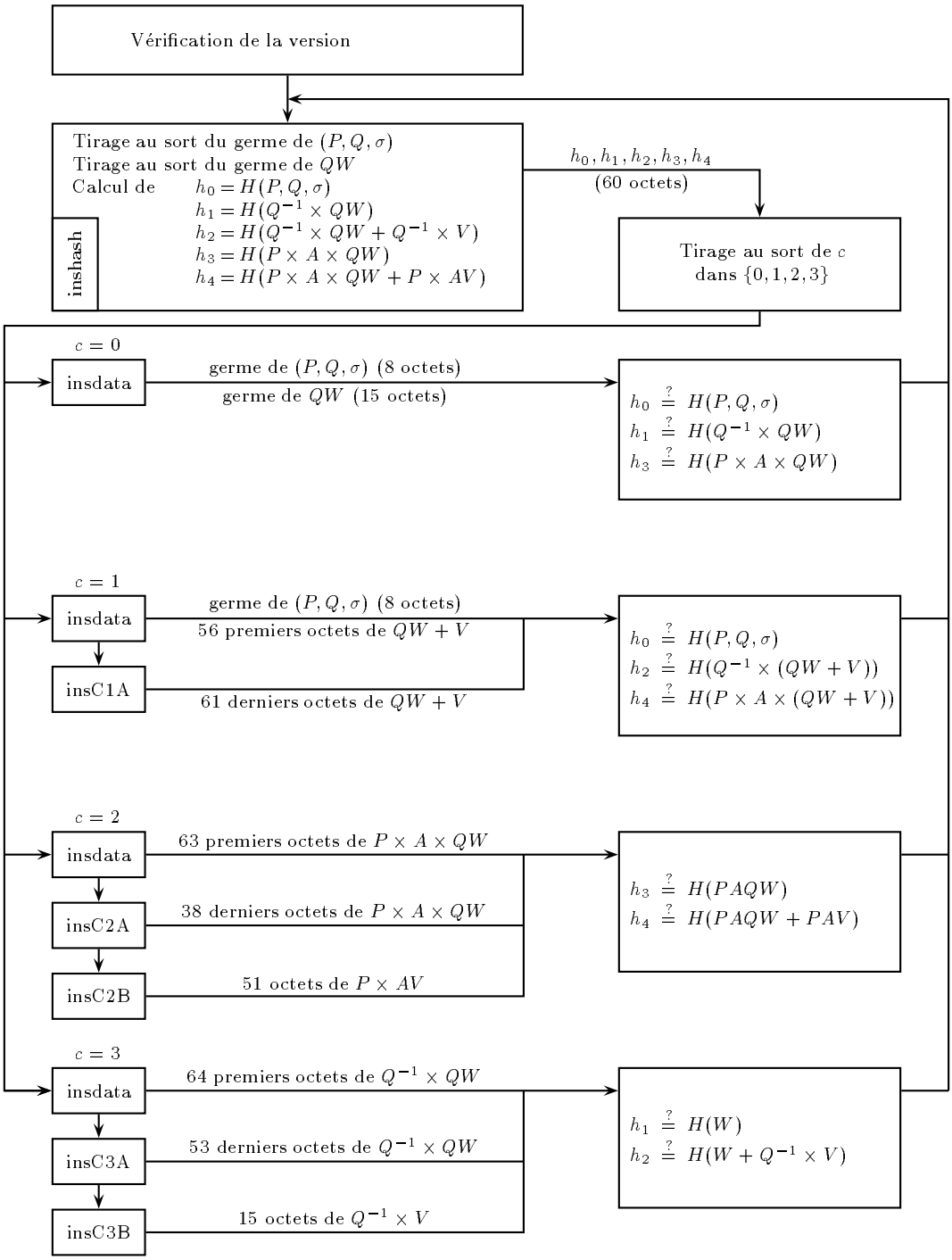

Fig. 4.10 – *Structure du protocole* ***PPP*** *à trois passes sur carte*

4.8.4 Performances

Les performances obtenues avec un boîtier cadencé à 3,57 MHz et un serveur géré par un IBM PS/2 80386 à 16 MHz sont regroupées dans le tableau de la figure 4.11. Tous ces temps représentent une authentification complète, c'est-à-dire 48 itérations, nécessaires pour obtenir une sécurité de un pour un million.

Protocole		moyenne	$c = 0$	$c = 1$	$c = 2$	$c = 3$
données (octets)	transférées	**9321**	4761	10041	11721	10761
	théoriques	8064	3984	8880	10176	9216
temps total (secondes) avec calculs serveur	à 19200 bauds	**53**	42	55	62	52
	à 9600 bauds	64	48	67	76	65
temps (secondes) sans calculs serveur	à 19200 bauds	43	29	42	55	45
	à 9600 bauds	55	35	54	68	59
calculs Carte à 3,57 MHz	cycles (10^6)	67,5	58,9	59,6	88,5	63
	secondes	18,9	16,5	16,7	25	17,6
calculs du serveur (secondes)		10	13	13	7	6
transferts (secondes) à 19200 bauds	réels	11	6	12	13	14
	théoriques	3,9	1,9	4,2	4,9	4,5
procédures de transfert (secondes)		13	6,5	16	15	14

Fig. 4.11 – *Performances de l'implantation*

Grâce à l'émulateur, nous avons pu détailler les temps de calcul, et remarquer que près de 40 % du temps est passé dans le hachage et près de 35 % dans le calcul des produits matrice/vecteur.

On constate qu'il faut alors entre 1 seconde et 1,5 seconde par tour. Cela convient parfaitement à une utilisation de contrôle d'accès en tache de fond, comme, par exemple, pour la télévision à péage. En effet, avec un test toutes les 10 secondes, un fraudeur a environ une chance sur six de voir une minute de film, et bien moins d'une sur un million d'en voir une heure.

4.8.5 Performances envisageables

De nombreuses améliorations sont envisageables avec un matériel légèrement plus performant :

- grâce à un émulateur de carte, nous avons pu tester une version plus efficace, mais un peu plus gourmande, du programme en assembleur. Il nécessite 2,5 Ko d'EEPROM (contre 2 Ko pour le programme effectif). Son temps de calcul baisse à 49,5 millions de cycles (au lieu de 67,5) ;

- il est tout à fait envisageable de faire fonctionner une carte à puce à 8 MHz au lieu de 3,57 MHz à l'aide d'un boîtier convenable. Ceci, combiné à l'amélioration précédente, amènerait le temps de calcul de la carte à 6 secondes au lieu de 18,9 actuellement ;

- les transmissions peuvent elles aussi être considérablement accélérées. Ainsi, à 115 200 bauds, valeur couramment utilisée, elles ne prendraient plus que 2 secondes ;
- le serveur que nous utilisons est particulièrement lent. Il est tout à fait concevable de ramener le temps de calcul de ce dernier à une durée de l'ordre d'une seconde en utilisant du matériel à peine plus performant ;
- en combinant l'augmentation des vitesses de la carte et du serveur, on peut enfin estimer la durée des procédures de transfert à 3 secondes.

Ces considérations nous amènent à la conclusion qu'il doit être possible, en utilisant une technologie actuellement courante et de faible coût, d'abaisser la durée d'une authentification complète à un temps de l'ordre de **12 secondes**.

Chapitre 5

Signatures électroniques

5.1 Définitions

La signature électronique (voir figure 5.1) a pour but de jouer le rôle, sur un document électronique, de la signature manuscrite sur un document papier : elle doit certifier à tout lecteur l'identité de l'auteur du document, mais elle doit également garantir toute personne contre la falsification de sa propre signature.

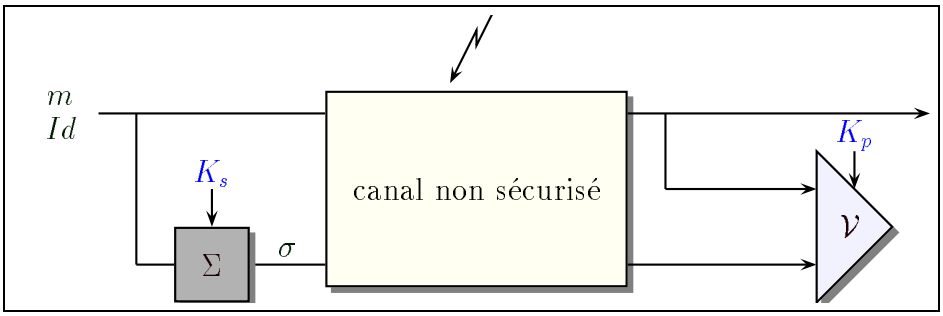

Fig. 5.1 – *Signatures électroniques*

Dans cette partie, nous allons donner une définition précise d'un schéma de signature, des attaques que nous envisagerons par la suite et de la sécurité d'un tel schéma ; en d'autres termes, nous allons définir ce que casser un schéma de signature signifie. Ces définitions sont inspirées de [50].

5.1.1 Éléments d'un schéma de signature

Dans un schéma de signature, chaque utilisateur révèle une clé publique et garde une clé secrète. La signature d'un utilisateur sur un message m est une valeur qui dépend de m et des clés secrète et publique de cet utilisateur. Quant à la validité de cette signature, elle peut être vérifiée par toute personne en possession de la clé publique. Il faut aussi qu'il soit difficile de contrefaire la signature d'un utilisateur sans connaître sa clé secrète.

Dans tous les schémas qui vont suivre (figure 5.2, figure 5.3, ...) apparaît un oracle f qui sera vu, en pratique, comme une fonction de hachage. Il aura son importance pour la

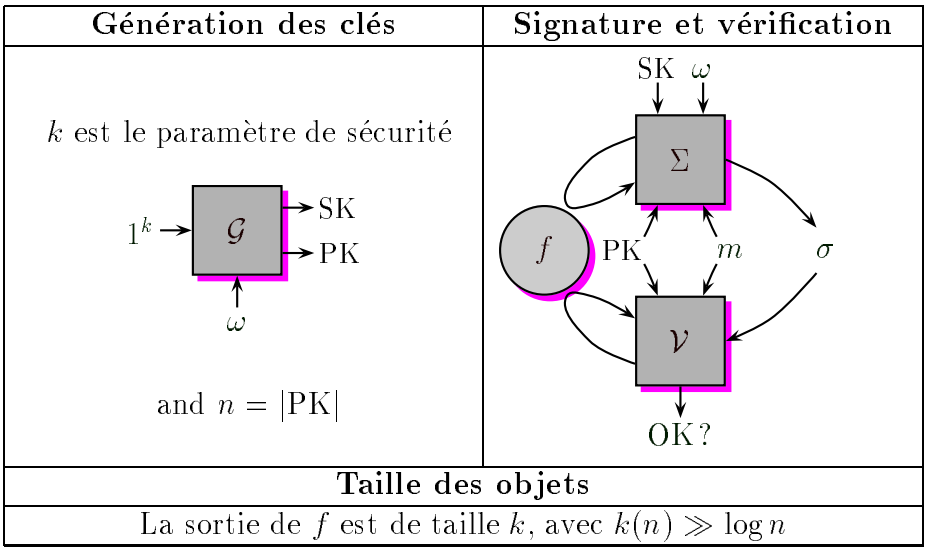

Fig. 5.2 – *Modélisation d'un schéma de signature*

famille de signatures que nous étudierons plus tard (paragraphe 5.3).

Définition 57. Un schéma de signature (voir figure 5.2) consiste en :

- Un *paramètre de sécurité* k, qui est choisi par l'utilisateur pour produire ses clés publique et secrète. Ce paramètre détermine la taille des clés, la sécurité générale, la longueur et le nombre maximal de messages qui pourront être signés, la taille de la signature, ainsi que le temps pris par l'algorithme de signature.

- Un *espace des messages* $M = \cup_k M_k$, où M_k est l'ensemble des messages pouvant être signés quand le paramètre de sécurité vaut k. Sans perte de généralité, on peut supposer que tous les messages sont représentés sous forme de chaînes binaires, c'est à dire que $M \subset \{0,1\}^*$. Pour s'assurer que le processus de signature en entier est polynomial, on supposera qu'il existe un entier c tel que la longueur des messages dans M_k soit majoré par k^c.

- Une *borne de sécurité* SB. La valeur $SB(k)$ représente le nombre maximal de messages qui peuvent être signés quand le paramètre de sécurité vaut k. À moins de précisions supplémentaires, $SB(k)$ sera un polynôme en k.

- Un *algorithme $\mathcal{G}$ de génération de clés* probabiliste et opérant en temps polynomial. $\mathcal{G}$ pourra être utilisé par chacun pour produire, à partir de la donnée 1^k, une paire (PK, SK) de clés publique et secrète. Soulignons ici que $\mathcal{G}$ doit être probabiliste, car la clé secrète produite par l'algorithme doit être imprévisible pour un adversaire.

- Un *algorithme de signature* Σ polynomial qui, à partir d'un message $m \in M_k$, des clés PK et SK, fournira une signature de m, $S = \sigma((PK, SK), m)$. Cet algorithme peut être probabiliste et peut aussi recevoir d'autres entrées.

– Un *algorithme de vérification* $\mathcal{V}$ fonctionnant en temps polynomial qui prendra en entrée une signature S, un message m, et une clé publique PK, et testera si S est une signature valide de m relativement à PK. En général, cet algorithme n'a nul besoin d'être probabiliste.

Pour un paramètre de sécurité donné, l'algorithme de génération des clés est utilisé de façon individuelle par chaque utilisateur pour obtenir ses propres clés. Les algorithmes de signature et de vérification sont communs pour les utilisateurs d'un même schéma de signature. Signalons enfin que, parfois l'ensemble des messages M_k peut dépendre de la clé publique PK.

5.1.2 Attaques

Par définition, nous distinguerons deux types d'attaques des fonctions de signature, les attaques sans message et les attaques par messages. Une attaque est dite *sans message*, si l'adversaire connaît uniquement la clé publique du signeur. En revanche, c'est une *attaque par messages* si l'adversaire peut aussi consulter des signatures de messages connus lorsqu'il veut casser la fonction.

Quatre types d'attaques par messages seront considérés selon le choix des messages dont la signature sera visible par l'adversaire. Dans la suite, Alice sera l'utilisatrice dont la fonction de signature sera attaquée.

Une attaque sera :

– *une attaque à messages connus*, si l'adversaire, Charlie, a accès à un ensemble de messages qu'il connaît mais qu'il n'a pas choisis.

– *une attaque à messages choisis générique*, si Charlie peut obtenir d'Alice les signatures d'un ensemble de messages qu'il choisit avant de pouvoir observer la clé publique d'Alice. Par conséquence, les messages sont construits par Charlie avant qu'il ait vu la moindre signature. Une telle attaque est appelée générique parce que ce sera la même attaque qui sera utilisée contre chaque signeur.

– *une attaque à messages choisis orientée*, si Charlie peut obtenir d'Alice les signatures d'un ensemble de messages qu'il choisit après avoir vu sa clé publique. Ici encore, tous les messages sont choisis avant d'avoir vu la moindre signature, mais l'attaque est dirigée contre Alice.

– *une attaque à messages choisis dynamique ou adaptative*, si Charlie peut obtenir d'Alice les signatures de messages qu'il choisit après avoir vu sa clé publique, et qui dépendent aussi des signatures déjà obtenues. Dans ce cas, Charlie peut utiliser Alice comme un oracle, et lui demander à tout moment de signer un message.

Il est clair que les attaques ci-dessus sont classées par ordre de sévérité ; l'attaque à messages choisis adaptative étant l'attaque la plus sérieuse que l'on puisse envisager. En général, on veut avoir des fonctions de signature résistant même aux attaques à messages choisis adaptatives, car cela signifie qu'un utilisateur pourra signer n'importe quel message sans craindre de compromettre sa sécurité.

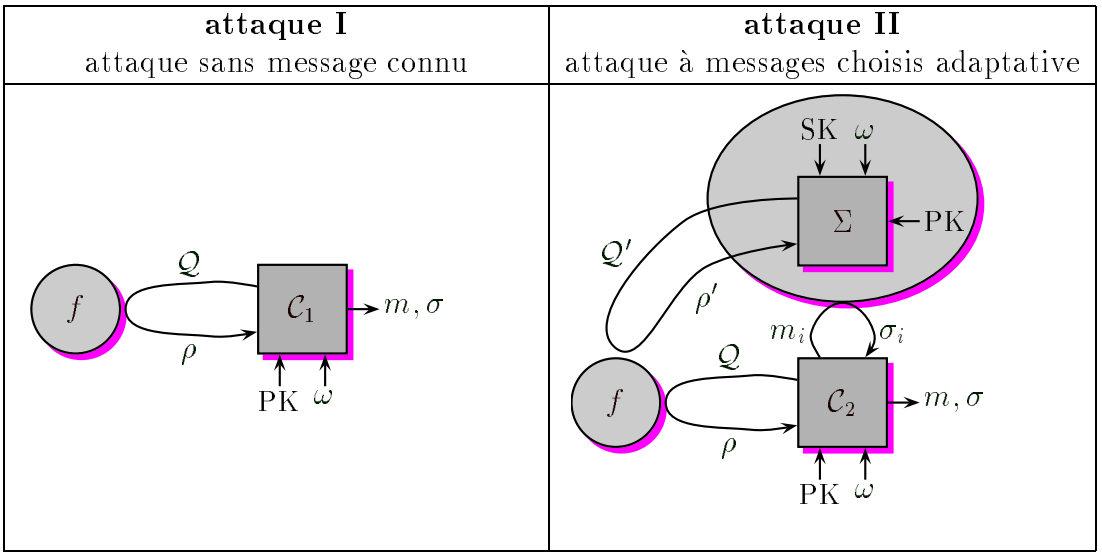

Fig. 5.3 – *Attaques d'un schéma de signature*

Dans les études qui vont suivre, nous ne considérerons que les deux extrêmes : les attaques sans message connu, car ce cas, bien entendu faible mais simple, permet de fournir des premiers résultats qui ensuite s'étendent, dans certains cas, de façon plus ou moins aisée aux attaques à messages choisis adaptatives (voir la figure 5.3).

5.1.3 Falsification et sécurité

Nous allons maintenant définir ce que signifie «casser» un schéma de signature. Charlie, le *faussaire*, sera vu comme une machine de Turing Polynomiale Probabiliste.

Soit $0 \leq f(k) \leq 1$, une fonction du paramètre de sécurité k et soit une attaque quelconque parmi celles précédemment mentionnées.

Nous dirons que Charlie $f(k)$-*casse* la fonction de signature d'Alice en utilisant l'attaque choisie ci-dessus, si avec probabilité au moins $f(k)$ (la probabilité est prise sur l'ensemble des tirages aléatoires de Charlie, des clés publiques et sur l'ensemble des signatures si l'algorithme de signature est probabiliste), il peut calculer une des quantités suivantes :

- la clé secrète d'Alice. Il s'agira alors d'un *cassage total*.
- une procédure efficace qui pourra simuler l'algorithme de signature d'Alice. On parlera de *falsification universelle*.
- la signature d'Alice pour au moins un message qu'il aura choisi. Il s'agira d'une *falsification sélective*.
- la signature d'Alice pour au moins un message. On parlera de *falsification existentielle*.

Précisons que pour Charlie, calculer une signature veut dire en donner une *nouvelle* qui n'aurait pas déjà été fournie par Alice.

Conformément aux définitions données ci-dessus, nous dirons qu'une fonction de signature est *totalement $f(k)$-cassable*, *universellement $f(k)$-falsifiable*, *sélectivement $f(k)$-falsifiable*, *existentiellement $f(k)$-falsifiable* s'il existe un faussaire qui la $f(k)$-casse dans le sens correspondant. Si $f(k) = 1$ pour tous les k, on parlera de fonction *totalement cassable*, *universellement falsifiable*, *sélectivement falsifiable*, *existentiellement falsifiable*. Ici les façons de casser sont données dans un ordre d'importance décroissante ; le moins qu'un adversaire puisse espérer étant une falsification existentielle avec une probabilité non négligeable.

Définition 58 (Négligeabilité). La fonction $f(k)$ est *négligeable*, si pour tout $c > 0$, pour tout k suffisamment grand, on a $f(k) < 1/k^c$.

Définition 59 (Schéma de signature sûr). Nous dirons qu'un schéma de signature est **sûr**, si même en utilisant une attaque à messages choisis adaptative, une falsification existentielle n'est possible qu'avec une probabilité négligeable.

5.2 Paradigme des fonctions à sens unique à trappe

5.2.1 Définition

Comme nous l'avons vu dans le chapitre d'introduction, il existe deux grandes familles de schémas de signature. La première utilise la propriété des permutations à sens unique à trappe. Pour cela, supposons que l'on possède une fonction f à sens unique, donc calculatoirement non inversible pour qui ne connaît pas la trappe g. Alice est alors la seule à connaître cette trappe g, et publie f telles que $f(g(x)) = x$. La fonction de hachage H est publique. Pour signer un message m, Alice calcule $\sigma = g(H(m))$. Elle envoie le message m accompagné de la signature σ. Bob peut alors vérifier que $f(\sigma) = H(m)$.

5.2.2 Exemples classiques

5.2.2.1 Signature RSA

L'exemple le plus classique est le schéma RSA [97] présenté en introduction, il utilise la fonction à sens unique à trappe $f(x) = x^e \mod N$, dont l'inverse est $g(x) = x^s \mod N$, où $es = 1 \mod \varphi(N)$.

- Alice possède une clé publique (N, e) et une clé secrète s. La fonction de hachage H est commune à tous les utilisateurs.
- Pour signer un message m, Alice calcule $\sigma = H(m)^s \mod N$. Elle envoie alors le couple (m, σ).
- Bob, mais également toute autre personne, peut vérifier, à l'aide de la clé publique (N, e) d'Alice, que $\sigma^e = H(m) \mod N$.

5.2.2.2 Signature de Rabin

Une autre fonction à trappe bien connue est le carré modulo un entier composé. Ce n'est pas exactement une permutation, car pour $N = pq$, le produit de deux grands premiers, les éléments de $(\mathbb{Z}/N\mathbb{Z})^*$ ont exactement quatre racines carrées ou aucune. Et donc, un élément sur quatre seulement admet une racine. L'application carré est simple à calculer, en revanche, sa réciproque est équivalente à la factorisation. Connaître les facteurs p et q de N permet de calculer aisément les racines carrées des résidus quadratiques. Michael O. Rabin [92, 93] a eu l'idée d'utiliser cette fonction à sens unique à trappe pour construire un schéma de signature :

- Alice calcule N, le produit de deux grands premiers p et q. Puis elle choisit un élément aléatoire, b, de $(\mathbb{Z}/N\mathbb{Z})^*$. Le couple (N, b) constitue sa clé publique et les facteurs (p, q) sa clé secrète. La fonction de hachage H est commune à tout utilisateur.

- Pour signer un message m, Alice concatène un mot aléatoire U de longueur fixée au message m et calcule $c = H(m||U)$. Puis elle teste si l'équation congruentielle $x(x + b) = c \mod N$ admet une solution. Si c'est le cas, elle en calcule une solution x, sinon, elle choisit un nouveau mot aléatoire U. En moyenne, elle aura quatre tentatives à effectuer pour obtenir une solution.

- Elle envoie alors le message m avec la signature (U, x).

- Bob, mais également toute autre personne, peut vérifier, à l'aide de la clé publique (N, b) d'Alice, que $x(x + b) = H(m||U) \mod N$.

5.2.3 Sécurité

Dans leurs différents articles [3, 4], Bellare et Rogaway ont prouvé que cette famille de schémas était sûre dans le modèle de l'oracle aléatoire. En effet, nous avons le résultat suivant, où H est vu comme un oracle aléatoire :

> ***Théorème 60 .***
> *Un cassage existentiel de la signature Σ utilisant la fonction à sens unique à trappe f selon une attaque à messages choisis adaptative dans le modèle de l'oracle aléatoire est équivalent à l'inversion de f.*

Preuve. Soit Charlie, noté $\mathcal{C}$, un attaquant capable de fabriquer, en temps t, après q_H appels à l'oracle aléatoire et q_σ appels à l'oracle de signature, une nouvelle signature valide avec probabilité non négligeable ε. On supposera qu'une question m posée à l'oracle de signature aura été préalablement posée à l'oracle aléatoire. On va alors construire une machine $\mathcal{M}$ capable d'inverser f en temps peu différent de t, avec probabilité ε/q_H.

Soit y un élément quelconque dont on cherche un antécédent par f. $\mathcal{M}$ choisit aléatoirement $\beta \in \{0, \ldots, q_H\}$, puis initialise le compteur i à 0.

$\mathcal{M}$ fait tourner $\mathcal{C}$:

- $\mathcal{C}$ demande le condensé d'un message m.
  - Si pour tout $j \leq i$, $m \neq m_j$, alors
    - on incrémente i ;
    - si $i \neq \beta$, on pose $m_i = m$, on choisit r_i aléatoirement, puis on calcule et retourne $y_i = f(r_i)$;
    - sinon, on pose $m_\beta = m$ et on retourne y.
  - Sinon, $m = m_j$ et on retourne à nouveau y_j.
- $\mathcal{C}$ demande la signature d'un message m. Comme supposé, il existe $j \leq i$ tel que $m = m_j$.
  - si $j = \beta$, $\mathcal{M}$ arrête et répond `ECHEC` ;
  - sinon, on retourne r_j.
- $\mathcal{C}$ termine et retourne (m, σ).

D'après l'hypothèse de l'oracle aléatoire, $H(m)$ est parfaitement imprédictible si la question m n'a jamais été posée. Dans ce cas, la probabilité pour que $H(m) = f(\sigma)$ est négligeable. Par conséquent, avec une chance sur q_H, $m = m_\beta$ et alors $H(m) = y = f(\sigma)$. □

Plus précisément, on peut énoncer les théorèmes suivants au sujet des signatures RSA et de Rabin précédemment présentées utilisant une fonction de hachage :

Théorème 61 (Sécurité de la signature RSA).
Un cassage existentiel de la signature RSA selon une attaque à messages choisis adaptative dans le modèle de l'oracle aléatoire est équivalent au problème RSA.

Théorème 62 (Sécurité de la signature de Rabin).
Un cassage existentiel de la signature de Rabin selon une attaque à messages choisis adaptative dans le modèle de l'oracle aléatoire est équivalent à la factorisation.

5.3 Paradigme du Zero-Knowledge

5.3.1 Définition

La deuxième famille provient des transformations de schémas d'identification à divulgation nulle de connaissance comme expliqué par Feige, Fiat et Shamir [37]. La technique est simple, elle consiste à paralléliser l'identification (ou tout du moins prendre des paramètres tels que la sécurité soit assez importante), et à remplacer l'interaction avec le vérifieur par un hachage. En fait, comme le «challenge» est posé par la fonction de hachage, supposée aléatoire, il suffit que le schéma d'identification utilisé soit à divulgation nulle de connaissance vis-à-vis d'un vérifieur honnête. Pour les preuves, on se limitera aux **schémas d'identification à 3 passes à divulgation nulle de connaissance face à un vérifieur honnête**. La parallélisation d'un schéma d'identification à 3 passes à divulgation nulle de connaissance en fait partie. Un schéma d'identification générique est présenté sur la figure 5.4.

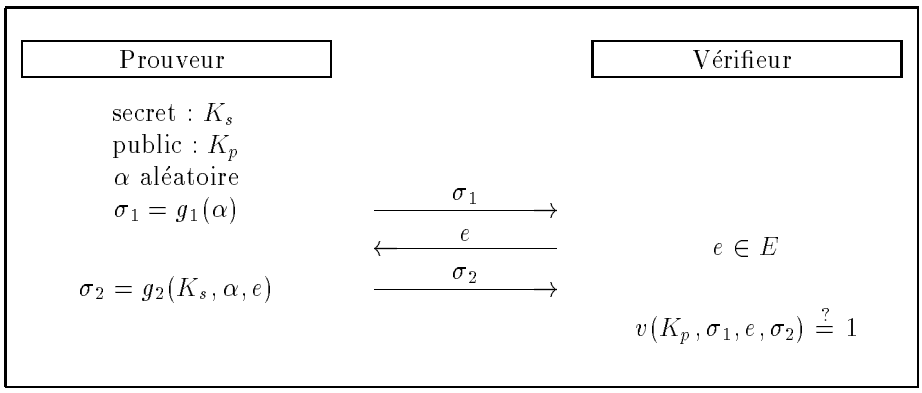

Fig. 5.4 – *Schéma d'identification à 3 passes générique*

Le schéma de signature associé est présenté figure 5.5, où m est le message à signer.

Définition 63. Un schéma de signature «à 3 passes» est un triplet $(\mathcal{G}, \Sigma, \mathcal{V})$ où

- $\mathcal{G}$ est l'algorithme de génération des clés (probabiliste) : sur le paramètre de sécurité k, $\mathcal{G}$ fournit un couple de clés publique et secrète. On notera n la taille de la clé publique.
- Σ est l'algorithme de signature (probabiliste) qui, à un message m, associe un couple (σ_1, σ_2) appelé signature.
- $\mathcal{V}$ est l'algorithme de vérification (déterministe) qui, sur un triplet (m, σ_1, σ_2), avec $f(m, \sigma_1)$, où f est une fonction de hachage, et à l'aide des données publiques π, répond 1 ou 0 selon que la signature est valide ou non.

Σ et $\mathcal{V}$ font appel à une fonction de hachage f, ou oracle aléatoire, qui fournit un condensé sur k bits. On supposera $k(n) \gg \log n$ tout en restant polynomial en n, la taille des clés. On peut alors remarquer qu'une fonction non négligeable devant k sera également non négligeable devant n. On abrégera alors en «non négligeable».

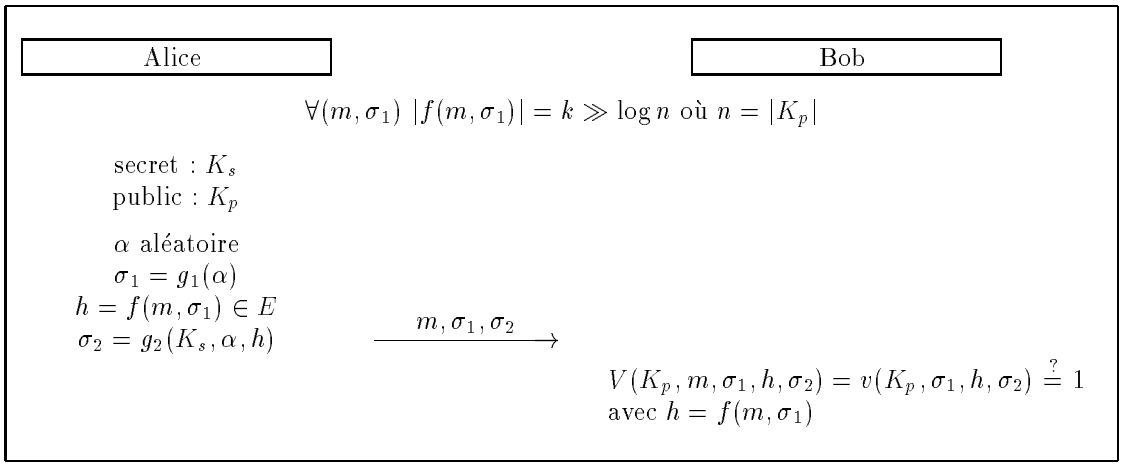

Fig. 5.5 – *Schéma de signature à 3 passes générique*

On supposera de plus que, pour tout (m,σ_1,σ_2), un unique «challenge» convient :

$$\Pr_h[V(\pi,k,m,\sigma_1,h,\sigma_2) = 1] \leq \frac{1}{2^k}.$$

En fait, ce type de schéma est assez général, il regroupe les principaux schémas de signature existants tels que les schémas de Fiat-Shamir [41], de Guillou-Quisquater [52] et de Schnorr [98, 99]. Le schéma de El Gamal [35] pourra être modifié en ce sens.

5.3.2 Exemple classique

L'exemple le plus classique et le plus efficace est le schéma de Schnorr [98, 99] qui repose sur la difficulté du logarithme discret dans un sous-groupe. Les données communes à tout utilisateur sont un grand premier p tel que $p-1$ admette un grand facteur premier q, ainsi qu'un élément α de $(\mathbb{Z}/p\mathbb{Z})^\star$ d'ordre q, puis une fonction de hachage f fournissant un condensé dans $(\mathbb{Z}/q\mathbb{Z})^\star$.

- Alice possède une clé secrète $s \in (\mathbb{Z}/q\mathbb{Z})^\star$ et la clé publique $v = \alpha^{-s} \bmod p$ associée.

- Pour signer un message m, Alice choisit un nombre aléatoire $r \in (\mathbb{Z}/q\mathbb{Z})^\star$ et calcule $x = \alpha^r \bmod p$. Le «challenge» est alors $e = f(m,x)$. Elle calcule $y = r + es \bmod q$. Elle envoie enfin le triplet (m,x,y).

- Bob, mais également toute autre personne, peut vérifier, à l'aide de la clé publique v d'Alice, que $x = \alpha^y v^e \bmod p$, où $e = f(m,x)$.

Remarque. Pour obtenir une signature plus courte, on envoie, en général, simplement le triplet (m,e,y). La vérification consiste alors à tester : $e \stackrel{?}{=} f(m,\alpha^y v^e \bmod p)$.

5.3.3 Techniques générales de preuve de sécurité

Dans cette partie, nous allons établir un théorème général, le *lemme de bifurcation*, permettant de prouver la sécurité des schémas de signature de cette famille. Cette technique a été présentée au congrès Eurocrypt '96 [86], par Jacques Stern et l'auteur de cette thèse, avec une application originale au schéma d'El Gamal [35].

La technique en question consiste à fabriquer une machine qui utilise l'attaquant Charlie afin de résoudre un problème difficile. Lors d'une attaque, Charlie, vu comme une Machine de Turing Polynomiale Probabiliste à Oracle $\mathcal{C}^f$, interroge l'oracle aléatoire f un nombre polynomial de fois avant de fournir une signature $(m, \sigma_1, h, \sigma_2)$, où $h = f(m, \sigma_1)$ et le triplet (m, σ_1, σ_2) est tel que défini dans la définition 63. Au sujet d'une telle attaque, nous définirons l'«indice» qui sera un élément primordial de l'analyse.

Notation 64. Nous appellerons «indice» la première fois où la question (m, σ_1) est posée à l'oracle, au cours de l'attaque, lorsque Charlie répond $(m, \sigma_1, h, \sigma_2)$. Cet indice vaut ∞ si cette question n'est jamais posée :

$$Ind(\omega, f) = \min\Big(\{i \,\big|\, \mathcal{Q}_i = (m, \sigma_1) \text{ où } \mathcal{C}^f(\omega) \longrightarrow (m, \sigma_1, h, \sigma_2)\} \cup \{\infty\}\Big).$$

Par analogie, nous avons

$$Ind(\omega, \rho) = \min\Big(\{i \,\big|\, \mathcal{Q}_i = (m, \sigma_1) \text{ où } \mathcal{C}^\rho(\omega) \longrightarrow (m, \sigma_1, h, \sigma_2)\} \cup \{\infty\}\Big).$$

Notre machine va donc faire tourner $\mathcal{C}^f(\omega)$ en espérant obtenir une signature valide avec un indice fini. On fait tourner à nouveau $\mathcal{C}^{f'}(\omega)$ avec un oracle aléatoire f' bien choisi, c'est-à-dire qui ne diffère de f qu'à partir de la question $\mathcal{Q}_{Ind}$. On espère encore une fois obtenir une signature valide avec le même indice que précédemment. Nous allons donc nous efforcer de prouver que la probabilité de succès de notre machine est non négligeable.

5.3.3.1 Attaques sans message connu

Nous allons commencer par les attaques sans message. Cette première version du «*lemme de bifurcation – sans message*» s'appliquera aux schémas dans lesquels la réponse à deux challenges distincts est supposée impossible pour un fraudeur, tels que les schémas de Fiat-Shamir [41] et de Schnorr [98, 99].

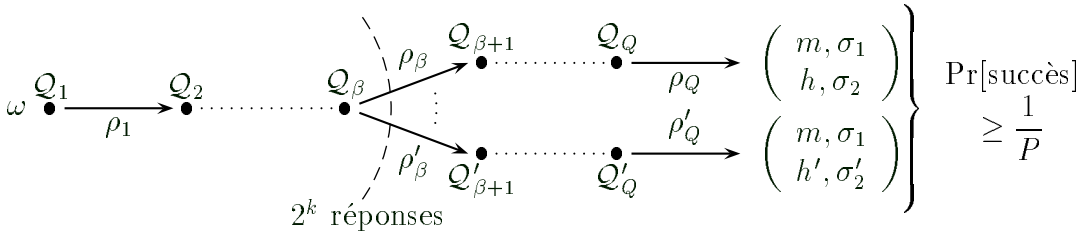

Fig. 5.6 – *Lemme de bifurcation*

Un résultat plus général, pour les schémas supposant la réponse à plus de γ challenges différents impossible, avec $\gamma > 2$, pourrait être énoncé de la même manière, et démontré de façon semblable. Cette extension permettrait de couvrir les schémas de signature fabriqués à partir de PKP [102], SD [112], CLE [113] ou PPP [83].

Énoncé

Théorème 65 (Lemme de bifurcation – sans message).
Supposons qu'une machine de Turing Polynomiale Probabiliste à oracle $\mathcal{C}^f$ puisse fabriquer des couples (message, signature) valides avec probabilité non négligeable.
Alors il existe une machine de Turing Polynomiale Probabiliste $\mathcal{M}$ capable de fournir deux signatures valides $(m, \sigma_1, h, \sigma_2)$ et $(m, \sigma_1, h', \sigma_2')$ telles que $h \neq h'$. Et ce, avec probabilité non négligeable.

Preuve

L'idée de la preuve est résumée sur la figure 5.6. Nous représenterons notre attaquant Charlie par la machine de Turing polynomiale probabiliste à oracle $\mathcal{C}^f(\omega)$. Cette machine fait $Q \geq 3$ appels à un oracle aléatoire f et, pour une infinité d'entiers n, obtient un succès, c'est-à-dire contrefait un couple (message, signature) valide $(m, \sigma_1, h, \sigma_2)$, avec probabilité non négligeable $\varepsilon \geq 1/P$, sur une fraction non négligeable de clés publiques $F(n)$:

$$\Pr_{\omega,f}[\mathcal{C}^f(\omega) \text{ succès}] = \Pr_{\omega,f}\left[\begin{array}{c} \mathcal{C}^f(\omega) \to (m, \sigma_1, h, \sigma_2) \text{ et} \\ V(\pi, k, m, \sigma_1, f(m, \sigma_1), \sigma_2) = 1 \end{array}\right] \geq \varepsilon.$$

Ici, comme tout au cours de cette thèse, P et Q sont des polynômes en n, et on rappelle que f fournit des condensés de taille $k \gg \log n$.

Regroupons dans l'ensemble $\mathcal{N}$ l'infinité d'entiers n pour lesquels $\mathcal{C}^f(\omega)$ est capable de contrefaire une signature valide pour toute clé publique de $F(n)$. Définissons le sous-ensemble $\mathcal{N}' = \{n \in \mathcal{N} \,|\, k(n) > \log(Q/\varepsilon) + 3\}$. Comme $\mathcal{N}$ est infini et $k(n) \gg \log n$, $\mathcal{N}'$ est également infini.

Soient $n \in \mathcal{N}'$ puis K_p une clé publique de $F(n)$. Comme il a été vu dans les préliminaires, on peut remplacer l'oracle f par un oracle ρ qui représente la pile des réponses :

$$\begin{aligned} \Pr_{\omega,f}[\mathcal{C}^f(\omega) \text{ succès}] &= \Pr_{\substack{\omega \\ \rho=(\rho_1,\ldots,\rho_Q)}}[\mathcal{C}^\rho(\omega) \text{ succès}] \\ &= \Pr_{\omega,\rho}\left[\begin{array}{c} \mathcal{C}^f(\omega) \to (m, \sigma_1, h, \sigma_2) \text{ et} \\ V(\pi, k, m, \sigma_1, \rho_\beta, \sigma_2) = 1 \end{array} \middle| \mathcal{Q}_\beta = (m, \sigma_1)\right] \geq \varepsilon. \end{aligned}$$

Éventuellement, $\beta = \infty$ et alors ρ_β est une nouvelle valeur aléatoire.

Cas où l'indice est égal à ∞. Commençons tout d'abord par montrer que Charlie pose, avec probabilité non négligeable, au cours de son attaque, la question (m, σ_1) à l'oracle avant de répondre $(m, \sigma_1, h, \sigma_2)$.

Lemme 66 .

$$\frac{\varepsilon}{Q+1} > \Pr_{\omega,f}[\mathcal{C}^f(\omega) \textit{ succès et } Ind(\omega, f) = \infty]$$

Preuve. Démontrons ce résultat par l'absurde. Supposons cette probabilité p_∞ supérieure à $\varepsilon/(Q+1)$. D'après ce qui a été vu dans les préliminaires,

$$p_\infty = \Pr_{\omega,\rho,h}\left[\begin{array}{l}\mathcal{Q}_i \neq (m,\sigma_1)\ \ \forall i \in \{1,\ldots,Q\}\\ V(\pi,k,m,\sigma_1,h,\sigma_2)=1\end{array}\ \middle|\ \mathcal{C}^\rho(\omega)\to(m,\sigma_1,h,\sigma_2)\right] \geq \varepsilon/(Q+1).$$

Donc d'après le lemme de séparation 28, il existe Ω tel que :

i) $\Pr_{\omega,\rho}[(\omega,\rho)\in\Omega] \geq \dfrac{\varepsilon}{2(Q+1)}$,

ii) $\forall(\omega,\rho)\in\Omega$, $\Pr_h\left[\begin{array}{l}\mathcal{Q}_i \neq (m,\sigma_1)\ \forall i \in \{1,\ldots,Q\}\\ V(\pi,k,m,\sigma_1,h,\sigma_2)=1\end{array}\ \middle|\ \mathcal{C}^\rho(\omega)\to(m,\sigma_1,h,\sigma_2)\right] \geq \dfrac{\varepsilon}{2(Q+1)}$.

Soient alors ω et ρ. Quitte à réitérer, on peut supposer $(\omega,\rho)\in\Omega$. Notons (m,σ_1,h,σ_2) la sortie de $\mathcal{C}^\rho(\omega)$. Ainsi,

$$\Pr_h\left[\begin{array}{l}\mathcal{Q}_i \neq (m,\sigma_1)\ \ \forall i \in \{1,\ldots,Q\}\\ V(\pi,k,m,\sigma_1,h,\sigma_2)=1\end{array}\right] \geq \frac{\varepsilon}{2(Q+1)}.$$

En particulier, $\Pr_h[V(\pi,k,m,\sigma_1,h,\sigma_2)=1] \geq \varepsilon/2(Q+1)$. Cependant, nous avons supposé $\Pr_h[V(\pi,k,m,\sigma_1,h,\sigma_2)=1] \leq 1/2^k$. En passant aux logarithmes, ceci se traduit par $k(n) \leq \log(2(Q+1)/\varepsilon)$. Ainsi, $k(n) \leq \log(Q/\varepsilon)+2$. Mais $n\in\mathcal{N}'$, ce qui implique $k(n) > \log(Q/\varepsilon)+3$, et entraîne une contradiction. □

Cas où l'indice est inférieur à Q. Nous savons donc désormais que $\mathcal{C}^f$ demande la valeur de $f(m,\sigma_1)$ à l'oracle avec probabilité non négligeable. En d'autres termes :

$$\Pr_{\omega,\rho}[\mathcal{C}^\rho(\omega) \text{ succès et } Ind(\omega,\rho)\leq Q] \geq \varepsilon\cdot Q/(Q+1).$$

Nous allons donc pouvoir utiliser cette machine en la faisant rejouer avec des oracles bien choisis pour fabriquer de bonnes signatures. En raison de la probabilité de succès, il existe $B\subseteq\{1,\ldots,Q\}$ non vide tel que

$$(\forall\beta\in B)\ \Pr_{\omega,\rho}[\mathcal{C}^\rho(\omega) \text{ succès et } Ind(\omega,\rho)=\beta] \geq \varepsilon/(Q+1).$$

Et, bien évidemment, $\Pr_\beta[\beta\in B]\geq 1/Q$. Supposons désormais que nous ayons un tel β.

On décompose maintenant les réponses de ρ en deux listes : la liste ρ_R des $\beta-1$ premières, et la liste ρ_S des suivantes. Par l'application du lemme de séparation 28 sur l'ensemble produit $\{(\omega,\rho_R)\}\times\{\rho_S\}$, il existe un ensemble Ω tel que :

i) $\Pr_{\omega,\rho_R}[(\omega,\rho_R)\in\Omega] \geq \dfrac{\varepsilon}{2(Q+1)}$,

ii) $\forall(\omega,\rho_R)\in\Omega$, $\Pr_\rho[\mathcal{C}^\rho(\omega) \text{ succès et } Ind(\omega,\rho)=\beta \,|\, \rho=(\rho_R,\rho_S)] \geq \varepsilon/2(Q+1)$.

Tirons au hasard ω et $(\rho_R, \rho_S) = \rho$. Avec probabilité supérieure à $(\varepsilon/2(Q+1))^2$, on a un couple (ω, ρ) qui mène au succès, avec $(\omega, \rho_R) \in \Omega$. Donc

$$\begin{aligned}
\frac{\varepsilon}{2(Q+1)} &\leq \Pr_{\rho'}[\mathcal{C}^{\rho'}(\omega) \text{ succès et } Ind(\omega, \rho') = \beta \,|\, \rho' = (\rho_R, \rho'_S)] \\
&\leq \Pr_{\rho'_S}[\mathcal{C}^{\rho'}(\omega) \text{ succès, } Ind(\omega, (\rho_R, \rho'_S)) = \beta \text{ et } \rho_\beta \neq \rho'_\beta] \\
&\quad + \Pr_{\rho'_S}[\mathcal{C}^{\rho'}(\omega) \text{ succès, } Ind(\omega, (\rho_R, \rho'_S)) = \beta \text{ et } \rho_\beta = \rho'_\beta] \\
&\leq \Pr_{\rho'_S}[\mathcal{C}^{\rho'}(\omega) \text{ succès, } Ind(\omega, (\rho_R, \rho'_S)) = \beta \text{ et } \rho_\beta \neq \rho'_\beta] + \Pr_{\rho'_S}[\rho_\beta = \rho'_\beta].
\end{aligned}$$

On en déduit

$$\Pr_{\rho'_S}[\mathcal{C}^{\rho'}(\omega) \text{ succès, } Ind(\omega, (\rho_R, \rho'_S)) = \beta \text{ et } \rho_\beta \neq \rho'_\beta] \geq \frac{\varepsilon}{2(Q+1)} - \frac{1}{2^k}.$$

Comme $n \in \mathcal{N}'$, on a $k > \log(Q/\varepsilon) + 2$, et donc $2^k \geq 4Q/\varepsilon$. En réinjectant dans l'inégalité ci-dessus, avec $Q \geq 3$, on obtient :

$$\Pr_{\rho'_S}[\mathcal{C}^{\rho'}(\omega) \text{ succès, } Ind(\omega, (\rho_R, \rho'_S)) = \beta \text{ et } \rho_\beta \neq \rho'_\beta] \geq \frac{\varepsilon}{12Q}.$$

Soit ρ'_S aléatoire : avec probabilité supérieure à $\varepsilon/12Q$, on a une deuxième signature. Les deux signatures satisfont $\mathcal{Q}_\beta = (m, \sigma_1) = (m', \sigma'_1)$. Si on pose $h = \rho_\beta$ et $h' = \rho'_\beta$, alors $h \neq h'$.

Nous avons désormais deux couples message–signature $(m, \sigma_1, h, \sigma_2)$ et $(m, \sigma_1, h', \sigma'_2)$ valides.

Conclusion La machine de Turing Polynomiale $\mathcal{M}$ suivante qui

- choisit aléatoirement l'indice de bifurcation β,
- choisit $\omega \in_R 2^\infty$, $\rho_R \in_R \left(\{0,1\}^k\right)^{\beta-1}$, $\rho_S \in_R \left(\{0,1\}^k\right)^{Q+1-\beta}$ et $\rho'_S \in_R \left(\{0,1\}^k\right)^{Q+1-\beta}$,
- pose $\rho = (\rho_R, \rho_S)$ et $\rho' = (\rho_R, \rho'_S)$,
- appelle $\mathcal{C}^{\rho}(\omega)$ et $\mathcal{C}^{\rho'}(\omega)$

fournit deux signatures $(m, \sigma_1, h, \sigma_2)$ et $(m, \sigma_1, h', \sigma'_2)$ avec $h \neq h'$, pour tout $n \in \mathcal{N}'$ tel que $Q(n) \geq 3$, avec probabilité supérieure à

$$\frac{1}{Q} \times \left(\frac{\varepsilon}{2(Q+1)}\right)^2 \times \frac{\varepsilon}{12Q} \geq \frac{1}{86} \times \frac{\varepsilon^3}{Q^4}.$$

Commentaires Nous avons alors un résultat asymptotique qui s'inscrit dans l'esprit de la théorie de la complexité. À la fin de ce chapitre, nous présenterons une réduction beaucoup plus efficace validant les schémas pratiques.

Ce théorème est un peu restrictif. En effet, son application est limitée aux protocoles pour lesquels deux réponses à deux challenges différents suite à une même mise en gage permettent d'extraire la clé secrète. Il concerne cependant les protocoles les plus connus : Fiat-Shamir, Schnorr, etc. Nous pouvons donc déjà prouver leur sécurité. Mais une preuve analogue fournit une réduction de l'ordre de

$$\frac{1}{Q} \times \left(\frac{\varepsilon}{2(Q+1)}\right)^{\gamma} \times \frac{\varepsilon}{12Q} \approx \frac{Cte}{Q} \times \left(\frac{\varepsilon}{Q}\right)^{\gamma+1}$$

d'une falsification existentielle en une attaque du problème de base pour tout schéma d'identification supposant la réponse à γ challenges distincts impossible sans la clé secrète.

Cette extension permet de prouver que toute transformation d'un protocole d'identification à 3 passes en un schéma de signature fournit un schéma de signature sans cassage existentiel possible selon une attaque sans message connu : les protocoles de Fiat-Shamir [41] (et ses variantes de Guillou-Quisquater [51], d'Otha-Okamoto [74], et d'Ong-Schnorr [78]), de Schnorr [98, 99], mais également de Shamir, PKP [102], de Stern, SD [112] et CLE [113], et **PPP** présenté au chapitre 4.

Comme nous l'avons vu précédemment, l'attaque sans message est l'attaque la plus faible. Ainsi, ce résultat de sécurité n'est pas très spectaculaire. Nous allons alors maintenant étendre cette preuve aux attaques par messages.

5.3.3.2 Attaques à messages choisis adaptatives

Nous allons généraliser le «lemme de bifurcation» afin de permettre son utilisation pour prouver la résistance aux attaques à messages choisis adaptatives.

Énoncé du théorème à messages choisis

Théorème 67 (Lemme de bifurcation – messages choisis).
Supposons qu'une machine de Turing Polynomiale Probabiliste à oracle $\mathcal{C}^{\Sigma^f, f}$ puisse fabriquer des couples (message, signature) valides avec probabilité non négligeable, en utilisant l'algorithme de signature Σ en oracle.
Supposons aussi que la combinaison de l'algorithme de signature Σ et de l'oracle aléatoire f puisse être simulée de façon indistinguable par une machine de Turing polynomiale probabiliste $\mathcal{S}$.
Alors il existe une machine de Turing Polynomiale Probabiliste $\mathcal{M}$ capable de fournir deux signatures valides $(m, \sigma_1, h, \sigma_2)$, et $(m, \sigma_1, h', \sigma_2')$ telles $h \neq h'$. Et ce, avec probabilité non négligeable.

Preuve
Soit $\mathcal{C}^{\Sigma^f(\omega'),f}(\omega)$ une machine de Turing Polynomiale Probabiliste qui fait appel à un oracle aléatoire f ainsi qu'à l'algorithme de signature Σ^f, comme oracle qui fait lui-même appel à l'oracle aléatoire f. Cette machine obtient un succès, c'est-à-dire contrefait un couple (message, signature) valide $(m, \sigma_1, h, \sigma_2)$, avec probabilité non négligeable, pour une infinité d'entiers n et pour une fraction non négligeable de clés publiques $F(n)$:

$$\Pr_{\omega,\omega',f}[\mathcal{C}^{\Sigma^f(\omega'),f}(\omega) \text{ succès}] = \Pr_{\omega,\omega',f}\left[\begin{array}{c}\mathcal{C}^{\Sigma^f(\omega'),f}(\omega) \to (m,\sigma_1,h,\sigma_2)\\ \text{et } V(\pi,k,m,\sigma_1,h,\sigma_2)=1\end{array}\right] \geq \frac{1}{P(n)}.$$

On suppose que (Σ, f) peut être simulé de façon indistinguable, c'est-à-dire que pour tout message m à signer, la distance entre la distribution (σ_1, h, σ_2) simulée et la distribution des triplets fournis par le signeur légal est négligeable.

$$\sum_{\tau_1,\rho,\tau_2}\left|\begin{array}{l}\Pr_{f,\omega}\left[\begin{array}{c|c}(\sigma_1,h,\sigma_2) & \Sigma^f(\omega,K_s,K_p,m)\\ =(\tau_1,\rho,\tau_2) & \to(\sigma_1,h,\sigma_2)\end{array}\right]\\ -\Pr_{\omega}\left[\begin{array}{c|c}(\sigma_1,h,\sigma_2) & \mathcal{S}(\omega,K_p,m)\\ =(\tau_1,\rho,\tau_2) & \to(\sigma_1,h,\sigma_2)\end{array}\right]\end{array}\right| \leq 1/poly(k).$$

On peut supposer que $\mathcal{C}$ appelle l'algorithme de signature exactement R fois, sur les messages $m^1, \ldots, m^R$, et à chaque fois, Σ fait un appel à f, et retourne (σ_1^j, σ_2^j) tels que

$$V(\pi, k, m^j, \sigma_1^j, f(m^j, \sigma_1^j), \sigma_2^j) = 1.$$

De son côté, $\mathcal{C}$ fait exactement Q appels distincts à f, $\mathcal{Q}_1, \ldots, \mathcal{Q}_Q$ (on supposera d'une part que $\mathcal{C}$ ne pose pas à f une question déjà posée par Σ, car la réponse est contenue dans la signature renvoyée par Σ, et d'autre part que si $\mathcal{C}$ devait faire moins de Q appels à f, elle ferait des appels aléatoires afin d'atteindre le nombre Q). En revanche, il est possible que Σ pose à f une question déjà posée par $\mathcal{C}$, ou par lui-même.

- Quelle est la probabilité pour que $\Sigma^f(\omega')$ pose à f une question déjà posée par $\mathcal{C}$?

$$\begin{aligned} p_1 &= \Pr_{\omega,\omega',f}[\Sigma^f(\omega') \text{ pose une question déjà posée par } \mathcal{C}] \\ &= \Pr_{\omega,\omega',f}[(\exists i \in \{1,\ldots,Q\})(\exists j \in \{1,\ldots,R\})((m^j,\sigma_1^j) = \mathcal{Q}_i)] \\ &\leq \sum_{i=1}^{Q}\sum_{j=1}^{R} \Pr_{\omega,\omega',f}[((m^j,\sigma_1^j) = \mathcal{Q}_i)]. \end{aligned}$$

  Sachant que σ_1 consiste en la mise en gage dans le schéma d'identification de base, on peut considérer que c'est une valeur aléatoire de longueur supérieure à k. Alors $p_1 \leq QR/2^k$.

- Quelle est la probabilité pour que $\Sigma^f(\omega')$ pose à f une question qu'il a lui-même déjà posée ? Ceci peut arriver si $\mathcal{C}$ demande de signer plusieurs fois le même message. Le pire des cas est lorsque $\mathcal{C}$ demande R signatures du même message :

$$\begin{aligned} p_2 &= \Pr_{\omega,\omega',f}[\Sigma^f(\omega') \text{ pose une question qu'il a déjà posée}] \\ &= \Pr_{\omega,\omega',f}[(\exists i \neq j \in \{1,\ldots,R(n)\})((m^i,\sigma_1^i) = (m^j,\sigma_1^j))] \ \leq \ R^2/2^k. \end{aligned}$$

Par conséquent, la probabilité d'obtenir une collision de questions est inférieure à $p_1 + p_2$: pour n assez grand,

$$\Pr_{\omega,\omega',f}\left[\begin{array}{c}\mathcal{A}^{\Sigma^f(\omega'),f}(\omega)\text{ succès}\\ \text{et pas de collisions de questions}\end{array}\right] \geq \varepsilon - p_1 - p_2 \;\geq\; \varepsilon - R(R+Q)/2^k \;\geq\; \varepsilon/2.$$

On peut donc remplacer l'oracle f par un oracle qui fournit des valeurs aléatoires indépendantes, comme expliqué dans les préliminaires :

$$\Pr_{\substack{\omega,\omega'\\ \tilde\rho_1,\ldots,\tilde\rho_R\\ \rho_1,\ldots,\rho_Q}}\left[\begin{array}{c}\mathcal{A}^{\Sigma^{\tilde\rho_1,\ldots,\tilde\rho_R}(\omega'),\rho_1,\ldots,\rho_Q}(\omega)\text{ succès}\\ \text{et pas de collisions de questions}\end{array}\right] \geq \frac{\varepsilon}{2}$$

où $\rho_1,\ldots,\rho_Q$ sont les réponses fournies par f aux questions $(\mathcal{Q}_1,\ldots,\mathcal{Q}_Q)$ posées par Charlie $\mathcal{C}$, et $\tilde\rho_1,\ldots,\tilde\rho_R$ les réponses fournies par f aux questions $(\tilde{\mathcal{Q}}_1,\ldots,\tilde{\mathcal{Q}}_R)$ posées par le signeur Σ. En appliquant le lemme de séparation 28, il existe $A \subseteq 2^\infty \times \left(\{0,1\}^{k(n)}\right)^R$ tel que

i) $\Pr_{\substack{\omega'\\ \tilde\rho_1,\ldots,\tilde\rho_R}}[(\omega',\tilde\rho_1,\ldots,\tilde\rho_R)\in A] \geq \varepsilon/4,$

ii) $\forall(\omega',\tilde\rho_1,\ldots,\tilde\rho_R)\in A \quad \Pr_{\substack{\omega\\ \rho_1,\ldots,\rho_Q}}\left[\begin{array}{c}\mathcal{C}^{\Sigma^{\tilde\rho_1,\ldots,\tilde\rho_R}(\omega'),\rho_1,\ldots,\rho_Q}(\omega)\text{ succès}\\ \text{pas de collisions de questions}\end{array}\right] \geq \frac{\varepsilon}{4}.$

Supposons que l'algorithme de signature $\Sigma^f(\omega')$ (qui nécessite la connaissance de la clé secrète) et l'oracle aléatoire puissent être remplacés par une machine de Turing Polynomiale Probabiliste $\mathcal{S}(\omega')$ qui n'utilise que des données publiques. Définissons la machine de Turing $\mathcal{M}$ suivante qui :

- choisit l'indice de bifurcation β,
- choisit les aléas ω' et ω,
- choisit $\rho_R \in_R \left(\{0,1\}^{k(n)}\right)^{\beta-1}$, $\rho_S \in_R \left(\{0,1\}^{k(n)}\right)^{Q(n)+1-\beta}$ et $\rho'_S \in_R \left(\{0,1\}^{k(n)}\right)^{Q(n)+1-\beta}$
- pose $\rho = (\rho_R, \rho_S)$ et $\rho' = (\rho_R, \rho'_S)$,
- fait tourner $\mathcal{C}^{\mathcal{S}(\omega'),\rho}(\omega)$ et $\mathcal{C}^{\mathcal{S}(\omega'),\rho'}(\omega)$.

Si on note $\tilde\rho_1,\ldots,\tilde\rho_R$ les réponses supposées de l'oracle aléatoire aux questions de l'algorithme de signature, alors avec probabilité supérieure à $\varepsilon/4$, $(\omega',\tilde\rho_1,\ldots,\tilde\rho_R)\in A$. Une fois ceci réalisé, une démonstration identique au théorème 65 montrerait que pour tout $n \in \mathcal{N}'$ tel que $Q(n) \geq 3$, avec probabilité supérieure à

$$\frac{1}{Q} \times \left(\frac{\varepsilon}{8(Q+1)}\right)^2 \times \frac{\varepsilon}{12Q} \geq \frac{1}{1366} \times \frac{\varepsilon^3}{Q^4},$$

$\mathcal{M}$ fournit deux signatures valides (m,σ_1,h,σ_2) et $(m,\sigma_1,h',\sigma'_2)$ avec $h \neq h'$.

Conclusion
Ce théorème permet de prouver la résistance aux attaques dynamiques de tous les schémas de signatures provenant d'un schéma d'identification à 3 passes à divulgation nulle de connaissance face à un vérifieur honnête.

5.3.4 Conséquences

Nous allons expérimenter ces théorèmes sur les schémas classiques de signature. Tout d'abord, nous étudierons le schéma de Fiat-Shamir [41]. Ensuite, nous verrons la sécurité du schéma de signature de Schnorr [98, 99].

5.3.4.1 Protocole de Fiat-Shamir

Nous allons tout d'abord appliquer ces théorèmes au schéma de Fiat-Shamir [41] pour obtenir un résultat plus ou moins évoqué par les auteurs en 1986, puis pour démontrer de plus la résistance aux attaques adaptatives. Le schéma de signature découle immédiatement de la parallélisation du schéma d'identification (voir figure 5.7) :

Une autorité choisit un module RSA N, produit de deux grands premiers p et q. Elle le publie ainsi qu'une fonction de hachage f. L'entier k est le paramètre de sécurité. Il doit croître très rapidement devant $\log n$, où n est la taille de N.

Alice, de son côté, choisit un élément aléatoire S de $(\mathbb{Z}/N\mathbb{Z})^\star$, le garde secret et publie le carré $I = S^2 \bmod N$.

Soit m le message à signer. Alice choisit k éléments aléatoires r_i dans $(\mathbb{Z}/N\mathbb{Z})^\star$, pour un indice i variant de 1 à k. Elle calcule les carrés, $x_i = r_i^2 \bmod N$, pour tout i, et définit le multiplet $\sigma_1 = (r_1, \ldots, r_k)$. Elle évalue $b_1 \ldots b_k = f(m, \sigma_1)$, puis calcule $y_i = r_i S^{b_i} \bmod N$ pour tout i. Elle définit le deuxième multiplet $\sigma_2 = (y_1, \ldots, y_k)$. Elle envoie alors le message m accompagné de la signature $\Sigma = (\sigma_1, \sigma_2)$.

Pour valider cette signature, Bob calcule le condensé $b_1 \ldots b_k = f(m, \sigma_1)$ et vérifie si $x_i = y_i^2 I^{-b_i} \bmod N$, pour tout $i \in \{1, \ldots, k\}$.

Pour une signature plus courte, on peut ne transmettre que le couple $(b_1 \ldots b_k, \sigma_2)$. La vérification se limite alors au test $b_1 \ldots b_k \stackrel{?}{=} f(m, (y_1^2 I^{-b_1} \bmod N, \ldots, y_k^2 I^{-b_k} \bmod N))$.

- Initialisation
  $N = pq$ produit de 2 grands premiers
  secret $S \in (\mathbb{Z}/N\mathbb{Z})^\star$
  public $I = S^2 \bmod N$

- Signature
  - $r_1, \ldots, r_k \in (\mathbb{Z}/N\mathbb{Z})^\star$
  - $x_i = r_i^2 \bmod N$ pour $i = 1, \ldots, k$
  - $b_1 \ldots b_k = f(m, x_1, \ldots, x_k) \in \{0,1\}^k$
  - $y_i = r_i S^{b_i} \bmod N$ pour $i = 1, \ldots, k$
  - $\sigma_1 = (x_1, \ldots, x_k)$ et $\sigma_2 = (y_1, \ldots, y_k)$

- Vérification
  - $b_1 \ldots b_k = f(m, x_1, \ldots, x_k)$
  - $y_i^2 \stackrel{?}{=} x_i I^{b_i} \bmod N$ pour $i = 1, \ldots, k$

Fig. 5.7 – *Schéma de signature de Fiat-Shamir à un secret*

Sécurité face aux attaques sans message connu. Commençons par étudier les attaques sans message connu.

Théorème 68 .
Dans le modèle de l'oracle aléatoire, une «falsification existentielle», suivant une «attaque sans message connu», du schéma de signature dérivé du protocole de Fiat-Shamir est équivalente à la factorisation.

Preuve. D'après le *lemme de bifurcation* (théorème 65), il existe une machine de Turing Polynomiale Probabiliste $\mathcal{M}$ capable de fournir deux signatures valides $(m, \sigma_1, h, \sigma_2)$ et $(m, \sigma_1, h', \sigma_2')$ telles que $h \neq h'$, pour une infinité d'entiers n, pour tout couple (N, I), où N est un entier de taille n, et I un résidu quadratique modulo N.

Si on pose $b_1 \dots b_k = h$, $c_1 \dots c_k = h'$, puis $(x_1, \dots, x_k) = \sigma_1$, $(y_1, \dots, y_k) = \sigma_2$ et enfin $(z_1, \dots, z_k) = \sigma_2'$, alors $y_i^2 = x_i I^{b_i} \bmod N$ et $z_i^2 = x_i I^{c_i} \bmod N$, pour tout $i \in \{1, \dots, k\}$. De plus, la différence $h \neq h'$ implique l'existence d'un indice d tel que $b_d \neq c_d$, par exemple, $b_d = 0$ et $c_d = 1$. Par conséquent, $y_d^2 = x_d \bmod N$ et $z_d^2 = x_d I \bmod N$, et donc $(z_d/y_d)^2 = I \bmod N$. □

Pour factoriser un entier N, on choisit $S \in_R \mathbb{Z}/N\mathbb{Z}$ et on calcule $I = S^2 \bmod N$. On laisse la machine de Turing définie ci-dessus attaquer la clé publique I et en trouver une racine carrée S'. Avec une probabilité d'un demi, $\text{pgcd}(S - S', N) \in \{p, q\}$.

Sécurité face aux attaques à messages choisis dynamiques Étudions maintenant les attaques à messages choisis dynamiques. Ce schéma de Fiat-Shamir a la propriété supplémentaire d'être à témoin indistinguable. En effet, utiliser une racine S ou S' de I fournira exactement les mêmes signatures. Par conséquent il n'est nul besoin d'utiliser un simulateur qui ne connaît pas la clé secrète. Il suffit de faire interagir l'attaquant avec un signeur utilisant une racine S de I. En appliquant le *lemme de bifurcation* sur $\mathcal{C}^{\Sigma,f}$, on obtient une autre racine S' différente de $\pm S$.

Théorème 69 .
Dans le modèle de l'oracle aléatoire, une «falsification existentielle», selon une «attaque à messages choisis dynamique», du schéma de signature dérivé du protocole de Fiat-Shamir est équivalente à la factorisation.

5.3.4.2 Protocole de Schnorr

Ce schéma est présenté sur la figure 5.8. Une autorité choisit et publie deux grands premiers p et q, tels que q soit un facteur de $p - 1$, ainsi qu'un élément α de $(\mathbb{Z}/p\mathbb{Z})^\star$ d'ordre q. La fonction de hachage f est publique et fournit un condensé dans $(\mathbb{Z}/q\mathbb{Z})^\star$. On

- Initialisation
  p, q deux grands premiers tels que $q|(p-1)$
  α élément de $(\mathbb{Z}/p\mathbb{Z})^\star$ d'ordre q
  secret $s \in (\mathbb{Z}/q\mathbb{Z})^\star$
  public $v = \alpha^{-s} \bmod p$
- Signature
  - $r \in (\mathbb{Z}/q\mathbb{Z})^\star$
  - $x = \alpha^r \bmod p$
  - $e = f(m, x)$
  - $y = r + se \bmod q$
  - $\sigma_1 = x$ et $\sigma_2 = y$
- Vérification
  - $e = f(m, x)$
  - $x \stackrel{?}{=} \alpha^y v^e \bmod p$

Fig. 5.8 – *Schéma de signature de Schnorr*

suppose que $\log q \gg \log\log p = \log n$. Alice, de son côté, choisit un élément aléatoire s de $(\mathbb{Z}/q\mathbb{Z})^\star$ qu'elle garde secret, et publie $v = \alpha^{-s} \bmod p$.

Soit m le message à signer. Alice choisit un élément aléatoire r dans $(\mathbb{Z}/q\mathbb{Z})^\star$ et calcule $x = \alpha^r \bmod p$. Elle évalue le condensé $e = f(m, x)$ puis calcule $y = r + se \bmod q$. Elle envoie alors le message m accompagné de la signature $\Sigma = (x, y)$.

La vérification consiste en le test de l'égalité $x \stackrel{?}{=} \alpha^y v^{f(m,x)} \bmod p$.

Pour une signature plus courte, on peut n'envoyer que le couple (e, y). La vérification devient $e \stackrel{?}{=} f(m, \alpha^y v^e \bmod p)$.

Preuves de sécurité

Théorème 70 .
Dans le modèle de l'oracle aléatoire, une «falsification existentielle», suivant une «attaque sans message connu» ou suivant une «attaque à messages choisis dynamique», du schéma de signature dérivé du protocole de Schnorr est équivalente au logarithme discret dans le sous-groupe.

Preuve. Tout d'abord, nous devons montrer l'existence d'un simulateur indistinguable d'un véritable signeur. Pour cela, nous allons prouver le lemme suivant, où p, q, α, s et v

sont choisis comme indiqué ci-dessus.

Lemme 71 .
Les distributions

$$\delta = \left\{ (x,e,y) \;\middle|\; \begin{array}{c} r \in_R (\mathbb{Z}/q\mathbb{Z})^\star \\ e \in_R (\mathbb{Z}/q\mathbb{Z})^\star \\ x = \alpha^r \bmod p \\ y = r + se \bmod q \end{array} \right\} \text{ et } \delta' = \left\{ (x,e,y) \;\middle|\; \begin{array}{c} r \in_R \mathbb{Z}/q\mathbb{Z} \\ e \in_R (\mathbb{Z}/q\mathbb{Z})^\star \\ y = r \\ x = \alpha^r v^e \bmod p \end{array} \right\}$$

sont statistiquement indistinguables.

Preuve. Soient $\epsilon \in (\mathbb{Z}/p\mathbb{Z})^\star$, $\gamma \in \mathbb{Z}/q\mathbb{Z}$ et $\beta \in (\mathbb{Z}/q\mathbb{Z})^\star$ tels que $\alpha^\gamma v^\beta = \varepsilon \bmod p$

$$\begin{aligned} p(\varepsilon,\beta,\gamma) &= \Pr_{\delta}[(x,f,y) = (\epsilon,\beta,\gamma)] = \Pr_{r\neq 0,e}[\alpha^r = \varepsilon \text{ et } e = \beta \text{ et } r + se = \gamma] \\ &= \Pr_{r\neq 0,e}[e = \beta \text{ et } r = \gamma - s\beta \text{ et } \alpha^\gamma v^\beta = \epsilon] = \frac{1}{q-1}\frac{1}{q-1}. \end{aligned}$$

$$\begin{aligned} p'(\varepsilon,\beta,\gamma) &= \Pr_{\delta'}[(x,f,y) = (\epsilon,\beta,\gamma)] = \Pr_{r,e}[\varepsilon = x = \alpha^r v^e \text{ et } e = \beta \text{ et } y = r = \gamma] \\ &= \Pr_{r,e}[e = \beta \text{ et } r = \gamma \text{ et } \alpha^\gamma v^\beta = \epsilon] = \frac{1}{q-1}\frac{1}{q}. \end{aligned}$$

Alors $|p - p'| = 1/q(q-1)^2$. Ce qui entraîne, puisque $\log q \gg \log n$,

$$\Delta = \sum_{(\varepsilon,\beta,\gamma)} |p - p'| = \frac{1}{q-1} \ll \frac{1}{poly(n)}.$$

□

Par conséquent, le simulateur $\mathcal{S}$ suivant génère des signatures avec une distribution statistiquement indistinguable de celle produite par le véritable algorithme de signature. Pour signer le message m, $\mathcal{S}^\rho$

- pose $e = \rho$,
- choisit $r \in_R \mathbb{Z}/q\mathbb{Z}$,
- pose et renvoie $x = \alpha^r v^e \bmod p$ et $y = r$.

Ce second lemme permet de prouver la faible probabilité de collisions de questions.

Lemme 72 .
Pour n assez grand, pour tout couple (m,x),

$$\Pr_{\omega',f}[\Sigma^f(\omega') \textit{ pose la question } (m,x)] \leq 2^{-\frac{n}{2}}.$$

Preuve.

$$\begin{aligned}
\Pr_{\omega',f}\left[\begin{array}{l}\Sigma^f(\omega') \text{ pose}\\ \text{la question } (m,x)\end{array}\right] &= \Pr_{\omega',f}\left[(a,b)=(m,x)\,\middle|\,\begin{array}{l}\Sigma^f(\omega') \text{ pose}\\ \text{la question } (a,b)\end{array}\right]\\
&= \Pr_{\omega',f}\left[(a,b)=(m,x)\,\middle|\,\begin{array}{l}r \in_R (\mathbb{Z}/q\mathbb{Z})^\star\\ (a,b)=(m,\alpha^r \bmod p)\end{array}\right]\\
&= \Pr_{r\neq 0}[b=\alpha^r \bmod p] \;\leq\; \frac{1}{q-1} \;\leq\; 2^{-\frac{n}{2}}.
\end{aligned}$$

□

D'après le théorème 67, il existe une machine de Turing Polynomiale Probabiliste $\mathcal{M}$ capable de fournir deux signatures (m,σ_1,h,σ_2) et $(m,\sigma_1,h',\sigma_2')$ valides, avec $h \neq h'$, pour une infinité d'entiers n et pour tout couple (p,α), où p est un nombre premier de taille n, et α un élément de $(\mathbb{Z}/p\mathbb{Z})^\star$ d'ordre q, un grand facteur de $p-1$.

Soient p un nombre premier de taille n, α un élément de $(\mathbb{Z}/p\mathbb{Z})^\star$ d'ordre q, où q est un grand facteur premier de $p-1$. Soit v, un élément de $(\mathbb{Z}/p\mathbb{Z})^\star$. On cherche à déterminer s tel que $v=\alpha^{-s} \bmod p$.

Faisons alors tourner la machine $\mathcal{M}$. En notant $x=\sigma_1$, $y=\sigma_2$ et $z=\sigma_2'$, on obtient $x=\alpha^y v^h \bmod p$ et $x=\alpha^z v^{h'} \bmod p$. Si on pose $s=(y-z)(h-h')^{-1} \bmod q$:

$$\alpha^{-s}=\alpha^{(y-z)(h'-h)^{-1}}=(xv^{-h}x^{-1}v^{h'})^{(h'-h)^{-1}}=(v^{h'-h})^{(h'-h)^{-1}}=v \bmod p.$$

□

5.3.5 Le schéma de El Gamal

Le schéma original de El Gamal [35] a été proposé en 1985 sans preuve d'équivalence au logarithme discret ni au schéma de distribution de clés de Diffie-Hellman [31]. Nous allons alors modifier un tout petit peu le schéma original afin de pouvoir appliquer les théorèmes précédents. Nous allons ainsi démontrer l'impossibilité d'une falsification existentielle selon une attaque à messages choisis adaptative.

5.3.5.1 Schéma initial

Une autorité choisit et publie un grand premier p ainsi qu'un générateur g de $(\mathbb{Z}/p\mathbb{Z})^\star$. Alice, de son côté, choisit un élément aléatoire x de $\left(\mathbb{Z}/(p-1)\mathbb{Z}\right)^\star$ et publie $y=g^x \bmod p$.

Soit m le message à signer. Alice choisit $K \in \left(\mathbb{Z}/(p-1)\mathbb{Z}\right)^\star$, calcule $r=g^K \bmod p$ et résout l'équation en s, $m=xr+Ks \bmod p-1$. Elle envoie le message m accompagné de la signature $\Sigma=(r,s)$.

Pour la vérification, on teste si $g^m=y^r r^s \bmod p$.

Résultats de sécurité

Théorème 73 ([35]).
Le schéma de signature de El Gamal est existentiellement falsifiable selon une attaque sans message connu.

Preuve. Soient $e \in_R \mathbb{Z}/(p-1)\mathbb{Z}$ et $v \in (\mathbb{Z}/(p-1)\mathbb{Z})^\star$, on pose $r = g^e y^v \bmod p$ puis $s = -rv^{-1} \bmod p-1$. Alors (r,s) est la signature du message $m = es \bmod p-1$:

$$g^m = g^{es} = y^r y^{sv} g^{es} = y^r (g^e y^v)^s = y^r r^s \bmod p.$$

□

5.3.5.2 Schéma modifié : El Gamal Modifié

Une autorité choisit et publie un grand premier p, un générateur g de $(\mathbb{Z}/p\mathbb{Z})^\star$ et **une fonction de hachage f** qui fournit un condensé de longueur $k \gg \log n$, où $n = \log p$. Alice, de son côté, choisit un élément aléatoire x de $(\mathbb{Z}/(p-1)\mathbb{Z})^\star$ et publie $y = g^x \bmod p$.

Soit m le message à signer. Alice choisit $K \in (\mathbb{Z}/(p-1)\mathbb{Z})^\star$, calcule $r = g^K \bmod p$ et résout l'équation en s, $\boldsymbol{f(m,r) = xr + Ks \bmod p-1}$. Elle envoie le message m accompagné de la signature $\Sigma = (r,s)$.

Pour la vérification, on teste si $g^{f(m,r)} = y^r r^s \bmod p$.

Résultats de sécurité Avant d'énoncer un quelconque résultat de sécurité sur ce schéma d'El Gamal modifié, nous allons définir une classe de familles de nombres premiers nécessaire pour une preuve au sens de la théorie de la complexité.

Définition 74. Un premier α-dur est un entier premier p tel que la factorisation de $p-1$ vérifie $p-1 = QR$ avec Q premier et $R \leq |p|^\alpha = n^\alpha$.

Attaques sans message connu

Théorème 75 .
Pour tout réel positif α, dans le modèle de l'oracle aléatoire, une «falsification existentielle», selon une «attaque sans message connu», du schéma de signature de El Gamal Modifié pour tout premier α-dur est équivalente au logarithme discret modulo les entiers premiers α-durs.

Preuve du théorème. D'après le *lemme de bifurcation* (théorème 65), il existe une machine de Turing Polynomiale Probabiliste $\mathcal{M}$ capable de fournir 2 signatures valides $(m, \sigma_1, h, \sigma_2)$ et $(m, \sigma_1, h', \sigma_2')$ avec $h \neq h'$, pour une infinité d'entiers n, pour tout entier α-dur p et pour une fraction non négligeable de couples (g,y). Soient $p = QR+1$ un premier α-dur de taille n et g un générateur de $(\mathbb{Z}/p\mathbb{Z})^\star$. Soit y un élément de $(\mathbb{Z}/p\mathbb{Z})^\star$. On cherche à déterminer x tel que $y = g^x \bmod p$. On fait fonctionner la machine $\mathcal{M}$ qui fournit deux quadruplets (m,r,h,s) et (m,r,h',t) tels que $g^h = y^r r^s \bmod p$ et $g^{h'} = y^r r^t \bmod p$ avec $h \neq h' \bmod p-1$. Alors,

$$g^{h-h'} = r^{s-t} \bmod p \quad (5.1)$$
$$g^{ht-h's} = y^{r(t-s)} \bmod p \quad (5.2)$$

Posons $a = r(s-t) \bmod p-1$.

Lemme 76 .
Les cas $pgcd(s-t, p-1) > R$ peuvent être exclus.

Preuve. $\gcd(s-t, p-1) > R$ signifie $s-t = 0 \bmod Q$, alors $h - h' = 0 \bmod Q$. Mais il ne faut pas oublier que dans le modèle de l'oracle aléatoire, h et h' sont des valeurs aléatoires. Par conséquent, la probabilité pour que $h' - h = 0 \bmod Q$ est inférieure à $1/Q$, donc négligeable. □

Il est donc clair, d'après (5.1), que $s-t$ est également premier avec Q. Quant à r, deux cas peuvent se présenter :

premier cas : $\gcd(r, Q) = 1$.
Alors, a est inversible modulo Q. En définissant $w = (sh' - th)a^{-1} \bmod Q$, si on énumère toutes les valeurs possibles de w modulo R, ce qui représente une quantité polynomiale, d'après (5.2), l'une d'elle fournira x.

deuxième cas : $\gcd(r, Q) = Q$.
Alors, il existe b et u tels que $r = bQ = g^u \bmod p$. Nécessairement, d'après (5.1), $h - h' = u(s-t) \bmod p-1$, donc u vérifie $u = (h-h')(s-t)^{-1} \bmod Q$. Par une recherche exhaustive sur la valeur de u modulo R, on détermine totalement u tel que $g^u = r = bQ \bmod p$. De plus, on peut remarquer que $\gcd(u, Q) = 1$.

Nous avons donc construit une machine $\mathcal{M}'$ telle qu'il existe un polynôme P' satisfaisant

$$\Pr_{g,y,\omega}[\mathcal{M}'(g,y,\omega) \to x \text{ ou } \mathcal{M}'(g,y,\omega) \to (b,u)] \geq \frac{1}{P'}.$$

Deux cas se présentent à nouveau :

1. $\Pr_{g,y,\omega}[\mathcal{M}'(g,y,\omega) \to x] \geq 1/2P'$.
2. $\Pr_{g,y,\omega}[\mathcal{M}'(g,y,\omega) \to (b,u)] \geq 1/2P'$.

Dans ce deuxième cas, puisque le nombre de b possibles est inférieur à R, il existe b_0 tel que $\Pr_{g,y,\omega}[\mathcal{M}'(g,y,\omega) \to (b_0,u)] \geq 1/2P'R$. D'après le lemme de séparation 28, il existe un ensemble non négligeable Y tel que,

$$\text{pour tout } y \in Y, \Pr_{g,\omega}[\mathcal{M}'(g,y,\omega) \to (b_0,u)] \geq 1/4P'R.$$

De même, pour tout $y \in Y$, il existe un ensemble non négligeable G_y tel que, pour tout $g \in G_y$, $\Pr_{\omega}[\mathcal{M}'(g,y,\omega) \to (b_0,u)] \geq 1/8P'R$.

Soit un couple (g,y) quelconque. Avec probabilité non négligeable, $y \in Y$ et $g \in G_y$, puis $\mathcal{M}'$ fournit (b_0,u) tel que $g^u = b_0 Q \bmod p$. Pour v aléatoire, avec probabilité non négligeable, $yg^v \in G_y$ et $\mathcal{M}'$ fournit (b_o,u'), tel que $(yg^v)^{u'} = b_0 Q = g^u \bmod p$. Alors

$y^{u'} = g^{u-vu'} \bmod p$. On avait déjà remarqué que $\text{pgcd}(u', Q) = 1$, ce qui permet de déterminer x modulo Q. Une recherche exhaustive sur x modulo R fournit x modulo $p-1$.

On a ainsi, dans tous les cas, une machine $\mathcal{M}''$ et une fonction non négligeable δ telles que $\Pr_{g,y,\omega}[\mathcal{M}''(g, y, \omega) \to x] \geq \delta$.

Soient g et y. Avec u, v et ω aléatoire, $\mathcal{M}''(g^u, g^v y, \omega)$ fournit un élément w tel que $g^v y = g^{uw} \bmod p$, avec probabilité non négligeable. Alors $y = g^{uw-v} \bmod p$, d'où

$$x = uw - v \bmod p - 1.$$

□

Attaques à messages choisis

Théorème 77 .
Pour tout réel positif α, dans le modèle de l'oracle aléatoire, une «falsification existentielle», selon une «attaque à messages choisis dynamique», du schéma de signature de El Gamal Modifié pour tout premier α-dur est équivalente au logarithme discret modulo tout entier premier α-dur.

Preuve. Comme vu précédemment, la preuve de ce théorème repose sur la possibilité de simuler le signeur légal.

Lemme 78 .
L'algorithme de signature peut être simulé sans clé secrète par une machine de Turing Polynomiale Probabiliste $\mathcal{S}$.

Preuve. Soient p, un nombre premier α-dur, $p-1 = QR$ et g un générateur de $(\mathbb{Z}/p\mathbb{Z})^\star$. La simulation s'effectue en utilisant la technique de la falsification existentielle du schéma original pour la partie modulo Q. L'autre partie est simulée par recherche exhaustive. Soient x un élément quelconque de $\left(\mathbb{Z}/(p-1)\mathbb{Z}\right)^\star$ et $y = g^x \bmod p$. Le simulateur $\mathcal{S}$ tire $u \in_R \mathbb{Z}/Q\mathbb{Z}$ et $t \in_R (\mathbb{Z}/Q\mathbb{Z})^\star$, et pose $e = uR \bmod (p-1)$ puis $v = tR \bmod (p-1)$, conformément à la falsification existentielle. Puis il tire également $\ell \in_R (\mathbb{Z}/R\mathbb{Z})^\star$. Il pose $r = (g^e y^v) g^{Q\ell} \bmod p$. On recommence la simulation dans le cas, peu probable, où r n'est pas un générateur de $\left(\mathbb{Z}/(p-1)\mathbb{Z}\right)^\star$. Pour suivre la falsification existentielle dans le sous-groupe engendré par g^R, on veut $h = -erv^{-1} \bmod Q$ et $s = -rv^{-1} \bmod Q$. Quant à la partie modulo R, on la choisit aléatoirement, $m = h \bmod R$. Puis, par une recherche exhaustive, on recherche la partie de s modulo R telle que $g^h = y^r r^s \bmod p$. On peut alors remarquer que le triplet (r, h, s) est une signature valide de m si $h = f(m, r)$.

Alors les distributions

$$\delta = \left\{ (r,h,s) \;\middle|\; \begin{array}{c} K \in_R (\mathbb{Z}/(p-1)\mathbb{Z})^\star \\ h \in_R \mathbb{Z}/(p-1)\mathbb{Z} \\ r = g^K \bmod p \\ h = Ks + xr \bmod p-1 \end{array} \right\} \text{ et } \delta' = \left\{ (r,h,s) \;\middle|\; \begin{array}{c} u \in_R \mathbb{Z}/Q\mathbb{Z} \\ t \in_R (\mathbb{Z}/Q\mathbb{Z})^\star \\ \ell \in_R (\mathbb{Z}/R\mathbb{Z})^\star \\ m \in_R \mathbb{Z}/R\mathbb{Z} \\ (r,h,s) = \mathcal{S}(u,t,\ell,m) \end{array} \right\}$$

sont statistiquement indistinguables. En effet, soient $a \in (\mathbb{Z}/p\mathbb{Z})^\star$, $b,c \in \mathbb{Z}/(p-1)\mathbb{Z}$, tels que $g^b = a^c y^a \bmod p$. Définissons les probabilités $p_\delta(a,b,c) = \Pr_\delta[(r,h,s) = (a,b,c)]$ et $p_{\delta'}(a,b,c) = \Pr_{\delta'}[(r,h,s) = (a,b,c)]$. Tout d'abord, si a n'est pas un générateur de $(\mathbb{Z}/p\mathbb{Z})^\star$, le triplet (a,b,c) ne peut apparaître ni par une simulation, ni par une signature légale. Dans le cas contraire, $a = g^K \bmod p$ avec K premier avec $p-1$. Ce triplet apparaît exactement une fois parmi toutes les exécutions possibles de l'algorithme de signature. En revanche, l'obtention de ce triplet par simulation est plus délicate. Nous devons avoir $e + xv + Q\ell = K \bmod (p-1)$, et plus précisément $e + xv = K \bmod Q$ et $K = \ell Q \bmod R$. De plus, $er + vh = 0 \bmod Q$. Le système suivant se pose alors :

$$\left\{ \begin{array}{rcl} u + xt & = & K/R \bmod Q \\ au + bt & = & 0 \bmod Q. \end{array} \right.$$

Si $b \neq xa \bmod Q$, alors ce système admet une unique solution, l'unique exécution du simulateur qui mène au triplet (a,b,c). Dans le cas contraire, $a^c = g^{b-ax} \bmod p$, et donc nécessairement $c = 0 \bmod Q$. De plus, la deuxième équation du système se réduit à $a(u+xt) = aK/R = 0 \bmod Q$. Par conséquent $a = b = c = 0 \bmod Q$.

Pour tout choix de t, soit $Q-1$ choix distincts possibles, il existe un unique u satisfaisant $u + xt = K/R \bmod Q$. Il y a également un unique choix de ℓ satisfaisant $K = \ell Q \bmod R$. Quant à m, il vaut nécessairement $h \bmod R$. Il est clair que la distance entre ces deux distributions provient de ce dernier cas :

$$\begin{aligned} \Delta &= \sum_{(a,b,c)} \mid \Pr_\delta[(r,f,s) = (a,b,c)] - \Pr_{\delta'}[(r,f,s) = (a,b,c)] \mid \\ &= \sum_{a=b=0 \bmod Q} \left| \frac{1}{\varphi(p-1) \times (p-1)} - \frac{Q-1}{\varphi(p-1) \times (p-1)} \right| \\ &= \sum_{a=b=0 \bmod Q} \frac{Q-2}{\varphi(p-1) \times (p-1)} \leq R^2 \times \frac{Q-2}{QR \times (Q-1)\varphi(R)} \leq \frac{1}{Q} \times \frac{R}{\varphi(R)}. \end{aligned}$$

Or d'après la formule de Landau [62], $\varphi(N) \geq N/\log(2N) = N/(\log N + 1)$. Donc

$$\Delta \leq \frac{1 + \alpha \log n}{Q} \leq \frac{1 + \alpha \log n}{2^{n-2}} \ll \frac{1}{poly(n)}.$$ □

Par ailleurs, il est clair que pour tout couple (m, σ_1),

$$\Pr_{\omega,f}\left[\begin{array}{l} \Sigma^f(\omega') \text{ pose} \\ \text{la question } (m,\sigma_1) \end{array} \right] = 1/Q \leq n^\alpha/2^{n-1} \ll 1/poly(n).$$

Par conséquent, le théorème 67 permet de prouver la sécurité contre des attaques à messages choisis dynamiques. □

5.3.5.3 Précautions nécessaires

Dans la preuve de sécurité précédente, les probabilités sont prises sur (g, y, ω). Par conséquent, si g n'est plus choisi aléatoirement, nos résultats s'effondrent. En effet, nous montrons que l'ensemble des générateurs g procurant une faible sécurité est négligeable. Cependant, cet ensemble n'est pas vide. D'ailleurs, Daniel Bleichenbacher [6] a exhibé une famille de tels générateurs. En effet, il prouve dans son article que le schéma original de El Gamal, où l'on signe $h(m)$, est existentiellement falsifiable pour certains générateurs bien choisis. De plus, sa preuve s'adapte aisément à notre schéma modifié. En d'autres termes, l'autorité qui choisit les éléments p et g peut y cacher une trappe lui permettant de signer ensuite sous le nom de qui elle veut sans la connaissance de la clé secrète.

Théorème 79 .
Si g est lisse et divise $p-1$, il est possible de signer une fraction significative de messages sans la clé secrète.

Cette contradiction apparente est bien expliquée dans [114]. Afin d'éviter ce genre de trappe, il serait bon que l'autorité présente une preuve de son honnête construction de g, comme suggéré pour le module p dans DSS.

5.3.5.4 Pratique

En pratique, ce sont déjà les premiers α-durs que nous utilisons. En effet, en raison de l'attaque «pas de bébé, pas de géant» de Shanks [104] et de l'utilisation possible des restes chinois (voir Pohlig-Hellman [82]), $p-1$ doit posséder un grand facteur de l'ordre de 120 à 140 bits, tout en étant lui même sur 512 bits.

5.3.6 Réduction plus performante

Si nous voulons un énoncé plus quantitatif du *lemme de bifurcation* 67, en étudiant de plus près la réduction utilisée, nous pouvons affirmer le théorème suivant :

Théorème 80 (Lemme de bifurcation).
Supposons que la machine de Turing $\mathcal{C}$ soit capable d'effectuer une falsification existentielle selon une attaque à messages choisis adaptative en temps T, avec probabilité $\varepsilon \geq 1/P$, après Q appels à l'oracle aléatoire et R appels à l'oracle de signature. Si de plus $\varepsilon \geq 16(R+1)(R+Q)\cdot 2^{-k}$, où k est la taille des sorties de l'oracle aléatoire, alors il existe une machine capable de fournir deux signatures valides $(m, \sigma_1, h, \sigma_2)$ et $(m, \sigma_1, h', \sigma_2')$ telles que $h \neq h'$, en temps $T' = 2T$, avec probabilité $\varepsilon' \geq \varepsilon^3/1366Q^4$.

Ces réductions polynomiales n'ont qu'un intérêt asymptotique dans le sens de la théorie de la complexité en raison de leur coût important. Il est cependant possible d'améliorer

considérablement le coût de ces réductions pour les rendre utilisables en pratique. En effet, voici une nouvelle machine de Turing $\mathcal{M}'$ qui utilise le fraudeur Charlie et tente de fournir deux signatures valides.

Soit ε la probabilité de succès de $\mathcal{C}$. Décrivons la stratégie de $\mathcal{M}'$:

- Première étape
  1. $\mathcal{M}'$ initialise le compteur $c = 0$.
  2. $\mathcal{M}'$ choisit les aléas ω' et ω et l'oracle aléatoire ρ.
  3. $\mathcal{M}'$ fait tourner $\mathcal{C}^{\mathcal{S}^{(\omega'),\rho}}(\omega)$.
  4. Si elle n'obtient pas une signature valide avec un indice fini et si $c < 1/\varepsilon$, alors elle incrémente c et retourne en 2.
  5. Si $c > 1/\varepsilon$, $\mathcal{M}'$ retourne `Échec`, sinon, elle note $(m, \sigma_1, h, \sigma_2)$ le quadruplet obtenu, β l'indice et coupe ρ en deux morceaux : ρ_R, les $\beta - 1$ premières réponses, et ρ_S les suivantes.
- Deuxième étape
  1. $\mathcal{M}'$ réinitialise le compteur $c = 0$.
  2. $\mathcal{M}'$ choisit une deuxième série aléatoire de réponses ρ'_S, et pose $\rho' = (\rho_R, \rho'_S)$.
  3. $\mathcal{M}'$ fait tourner $\mathcal{C}^{\mathcal{S}^{(\omega'),\rho'}}(\omega)$ et incrémente c.
  4. Si elle n'obtient pas une signature valide avec l'indice égal à β et si $c < Q/\varepsilon$, alors elle retourne en 2.
  5. Si $c > Q/\varepsilon$, $\mathcal{M}'$ retourne `Échec`, sinon, elle note $(m, \sigma_1, h', \sigma'_2)$ le quadruplet obtenu.
- $\mathcal{M}'$ retourne les deux signatures $(m, \sigma_1, h, \sigma_2)$ et $(m, \sigma_1, h', \sigma'_2)$.

On peut déjà remarquer que le temps d'exécution T' de cette machine est borné par $(Q + 1)T/\varepsilon$. Quant à la probabilité de succès, on peut prouver qu'elle est supérieure à $1/30$. En effet, la probabilité d'obtenir le premier échec est égale à

$$\Pr[\mathcal{C}^{\mathcal{S}^{(\omega'),\rho}}(\omega) \text{ échoue } 1/\varepsilon \text{ fois}] = \Big(1 - \Pr[\mathcal{C}^{\mathcal{S}^{(\omega'),\rho}}(\omega) \text{ succès}]\Big)^{1/\varepsilon} \leq \Big(1 - \varepsilon/2\Big)^{1/\varepsilon} \leq 0,61.$$

Donc la première étape est franchie avec probabilité supérieure à $1/3$. Définissons

$$E = \left\{ (\beta, \omega, \omega', \rho_R) \,\middle|\, \Pr_{\rho_S}[\mathcal{C}^{\mathcal{S}^{(\omega'),\rho}}(\omega) \text{ succès et } Ind = \beta \,|\, \rho = (\rho_R, \rho_S)] \geq \varepsilon/4Q \right\}.$$

Par une démonstration similaire au lemme de succès 29, on montre que

$$\Pr_{\omega,\omega',\rho}[(\beta, \omega, \omega', \rho_R) \in E \,|\, \mathcal{C}^{\mathcal{S}^{(\omega'),\rho}}(\omega) \text{ succès}, \beta = Ind \text{ et } \rho = (\rho_R, \rho_S)] \geq 1/2.$$

Ainsi, avec probabilité supérieure à 1/6, la première étape est franchie avec la propriété supplémentaire que $(\beta, \omega, \omega', \rho_R) \in E$. Dans ce cas, la probabilité d'obtenir un échec à la deuxième étape vérifie

$$\Pr[\texttt{Échec}] = \Big(1 - \Pr[\mathcal{C}^{\mathcal{S}^{(\omega')},\rho}(\omega) \text{ succès et } Ind = \beta]\Big)^{Q/\varepsilon} \leq \Big(1 - \varepsilon/4Q\Big)^{Q/\varepsilon} \leq 0,78.$$

Donc cette dernière étape est franchie avec probabilité supérieure à 1/5. On peut alors énoncer le théorème suivant :

Théorème 81 (Lemme de bifurcation amélioré).
Supposons que la machine de Turing $\mathcal{C}$ soit capable d'effectuer une falsification existentielle selon une attaque à messages choisis adaptative en temps T, avec probabilité $\varepsilon \geq 1/P$, après Q appels à l'oracle aléatoire et R appels à l'oracle de signature. Si de plus $\varepsilon \geq 16(R+1)(R+Q) \cdot 2^{-k}$, où k est la taille des sorties de l'oracle aléatoire, alors il existe une machine capable de fournir deux signatures valides $(m, \sigma_1, h, \sigma_2)$ et $(m, \sigma_1, h', \sigma_2')$ telles que $h \neq h'$, en temps $T' \leq 2QT/\varepsilon$, avec probabilité $\varepsilon' \geq 1/30$.

Commentaire Ce *lemme de bifurcation amélioré* est optimal sur la contrainte imposée à ε, $\varepsilon \geq 16(R+1)(R+Q) \cdot 2^{-k}$. En effet, elle tient compte des probabilités de succès par chance ou hasard après $Q+R$ réponses de l'oracle aléatoire et de risque de collisions dues au simulateur.

En revanche, le facteur Q dans $T' \leq 2QT/\varepsilon$, qui provient du besoin d'obtenir deux fois le même «indice», est intrinsèque à notre réduction. Nous ne sommes pas parvenus à le supprimer comme Bellare et Rogaway [4] ont pu le faire pour les schémas de signature à base de fonctions à sens-unique à trappe. Ceci semble lié à la structure des schémas de signature selon le paradigme du «zero-knowledge».

Application numérique Ce nouveau résultat, tout en fournissant une réduction de complexité nettement moins élevée, est encore insuffisant pour une utilisation efficace en pratique. Cependant, nous pouvons essayer d'évaluer la sécurité du schéma de signature de Schnorr.

Notons τ_h et τ_σ les temps respectifs d'une question à l'oracle aléatoire et à l'oracle de signature. Nécessairement, $T \geq Q\tau_h + R\tau_\sigma$. Pour falsifier une signature, avec probabilité supérieure à 1/2, il faut faire fonctionner $\mathcal{C}$ environ $1/\varepsilon$ fois. Par conséquent, une fausse signature nécessite une durée $t = T/\varepsilon \geq Q\tau_h/\varepsilon$. Pour obtenir une «bifurcation» utile, il faut une durée d'attaque de l'ordre de $t' = 30T' \leq 60QT/\varepsilon \leq 60Tt/\tau_h = 60t^2\varepsilon/\tau_h \leq 60t^2/\tau_h$.

Supposons que τ_h représente l'équivalent de 2^m opérations. Bien entendu, t est limité à 2^{64} opérations, ainsi $t' \leq 2^{134-m}$.

La signature de Schnorr repose sur le logarithme discret dans un sous-groupe d'ordre premier q. Or, les meilleures attaques réalistes jusqu'à présent sont en racine de q («pas de bébé, pas de géant» [104]). On remarque alors qu'il suffit de prendre q sur, environ, 240 bits pour obtenir une signature sûre contre les attaques à messages choisis adaptatives.

Chapitre 6

Signatures en blanc

6.1 Définitions et exemples

6.1.1 Signatures en blanc

Dans cette partie, nous allons tout d'abord expliquer ce qu'est une signature en blanc, et à quoi elle peut servir, car cette notion est moins naturelle que la signature électronique. Nous verrons très rapidement quelques exemples, puis enfin la sécurité que l'on peut leur réclamer. Certains de ces résultats ont été présentés à Asiacrypt '96 [85] par Jacques Stern et l'auteur de cette thèse. Quant aux exemples équivalents à factorisation, ils seront présentés à la conférence «Computer and Communication Security» [87] de l'ACM.

Une signature en blanc, comme l'indique son nom, par analogie aux chèques en blanc, consiste à faire signer à une personne un document qui lui est inconnu. De plus, on souhaite que cette personne soit ensuite incapable de reconnaître cette signature et/ou d'y relier un message. En d'autres termes, après avoir ainsi signé plusieurs documents, le signeur est incapable, à la vue d'un message signé «par sa main», de déterminer à quelle signature en blanc ce message correspond.

La notion de signature en blanc permet donc typiquement d'assurer l'anonymat en monnaie électronique. Pour comprendre cela, nous allons quelque peu anticiper sur l'historique de la monnaie électronique qui sera présenté au chapitre suivant. Depuis son invention, la monnaie électronique a été concrétisée par un nombre certifié par la Banque. Ce nombre certifié sera alors appelé «pièce de monnaie». Pour garantir l'anonymat, cette pièce doit être inconnue de la Banque. Cette dernière procède donc à une signature en blanc sur une pièce que l'utilisateur crée de façon secrète. Nous verrons que ce concept est suffisant dans un contexte «on-line». Pour passer à la monnaie électronique «off-line», c'est-à-dire où le commerçant n'est pas constamment en communication avec la Banque, on devra imposer à la pièce une structure un peu plus complexe pour y introduire l'identité de l'utilisateur, afin de pouvoir le retrouver en cas de fraude. Pour être assurée de la présence de l'identité de l'utilisateur dans la pièce, la Banque utilisera la technique du «cut-and-choose». Cela consiste à choisir un échantillon aléatoire de pièces fabriquées et de demander à l'utilisateur de dévoiler leur contenu afin de le vérifier. Ces pièces n'étant plus anonymes, elles sont jetées. Mais cette technique étant coûteuse en terme de complexité de calculs et transfert de données, aussi, plus récemment, sont apparues des techniques

plus élégantes.

Cependant, quelle que soit la technique utilisée, il reste à voir si cette pièce signée en blanc permet à la Banque de garder le contrôle de la quantité d'argent en circulation. Pour cela, il faut qu'après avoir reçu n signatures en blanc, un utilisateur ne soit pas capable de fabriquer $n+1$ pièces. Cette notion de sécurité est, de façon surprenante, tout juste citée, et simplement supposée dans les derniers schémas de monnaie électronique [12] !

6.1.2 Exemples

Avant de définir les notions de sécurité, nous allons voir les principaux schémas de signature en blanc déjà présentés pour des protocoles de monnaie électronique. Dans ces exemple, H sera une fonction de hachage.

6.1.2.1 RSA en blanc

Nous allons tout d'abord présenter la transformation en blanc du schéma RSA [97]. Ce fut le schéma utilisé par David Chaum [19, 20, 21] dans les premiers systèmes de monnaie électronique.

Dans le contexte de RSA, nous avons un grand nombre composé $N = pq$, appelé module RSA, une clé publique ou clé de vérification, e entier inférieur à N, premier avec $\varphi(N) = (p-1)(q-1)$, et la clé secrète associée ou clé de signature, $s = e^{-1} \bmod \varphi(N)$. La signature d'un message m est la racine e-ième de $H(m)$, $\sigma = H(m)^{1/e} = H(m)^s \bmod N$.

Maintenant, pour obtenir la signature d'un message secret m, l'utilisateur «aveugle» ce message par une valeur aléatoire $r^e \bmod N$, et envoie alors $h' = H(m)r^e \bmod N$ au signeur. Ce dernier renvoie la signature σ' de h' telle que $\sigma'^e = h' = r^e H(m) \bmod N$. Il est aisé de remarquer que $\sigma = \sigma' r^{-1} \bmod N$ est une signature valide de m. Quant au signeur, il est totalement incapable de relier la signature qu'il a fournie σ' à la signature finale σ, même avec une puissance de calcul infinie.

6.1.2.2 Schnorr en blanc

Le schéma de signature de Schnorr [98, 99] peut également être converti en schéma de signature en blanc. Cette transformation fut utilisée dans les premiers schémas de monnaie électronique sans «cut-and-choose» [9, 8, 40, 39].

Nous avons alors deux grands entiers premiers p et q, tels que $q \mid p-1$. Ils sont publiés, ainsi qu'un élément g de $(\mathbb{Z}/p\mathbb{Z})^\star$ d'ordre q. Le signeur crée une paire de clés, la clé secrète $x \in \mathbb{Z}/q\mathbb{Z}$ et la clé publique $y = g^{-x} \bmod p$.

Pour une signature standard, le signeur choisit un nombre aléatoire $k \in \mathbb{Z}/q\mathbb{Z}$ et calcule $r = g^k \bmod p$. Le hachage du couple (m, r) lui fournit le «challenge» e. Il calcule alors $s = k + xe \bmod q$. On peut vérifier que $r = g^s y^{H(m,r)} \bmod p$ ou bien plus simplement $e = H(m, g^s y^e \bmod p)$.

Pour une signature en blanc, le signeur choisit un nombre aléatoire $k \in \mathbb{Z}/q\mathbb{Z}$ et envoie $r = g^k \bmod p$. L'utilisateur masque cette mise en gage par deux nombres aléatoires $\alpha, \beta \in \mathbb{Z}/q\mathbb{Z}$ en $r' = rg^{-\alpha} y^{-\beta} \bmod p$ et calcule $e' = H(m, r') \bmod q$. Il envoie alors le challenge $e = e' + \beta \bmod q$ au signeur qui lui retourne une valeur s vérifiant $g^s y^e = r \bmod p$. On peut aisément vérifier qu'avec $s' = s - \alpha \bmod q$, (e', s') est une signature valide du

message m puisque ce couple vérifie $e' = H(m, g^{s'} y^{e'} \bmod p)$. Lors de ce protocole, le message m n'a jamais été révélé et le triplet (m, r', s') est indépendant de la communication que l'utilisateur a eue avec la Banque.

Nous pouvons maintenant définir, de façon plus précise, des notions de sécurité pour les signatures en blanc.

6.2 Sécurité

Nous venons de voir l'application, pour le moment essentielle, en monnaie électronique des signatures en blanc. L'analyse de ce cas pratique nous conduit à définir des critères de sécurité souhaitables ou nécessaires. Pour ce qui est de l'anonymat, on aimerait, pour le moment, l'avoir de façon inconditionnelle, ce qui est assuré par l'aspect indistinguable des signatures.

Définition 82 (Indistinguabilité). Supposons que la Banque ait aidé à signer deux messages m et m', par σ et σ', suite aux interactions I et I'. Alors à la vue de (m, σ), la Banque ne peut distinguer, même avec une puissance infinie, la provenance de cette signature, entre les communications I et I'.

Nous pouvons alors définir plus précisément ce qu'est un schéma de signature en blanc.

Définition 83 (Signature en blanc). Un schéma de signature en blanc est un schéma de signature que l'on a rendu interactif permettant ainsi à quiconque de faire signer à une autorité un message, tout en lui faisant respecter la propriété d'indistinguabilité.

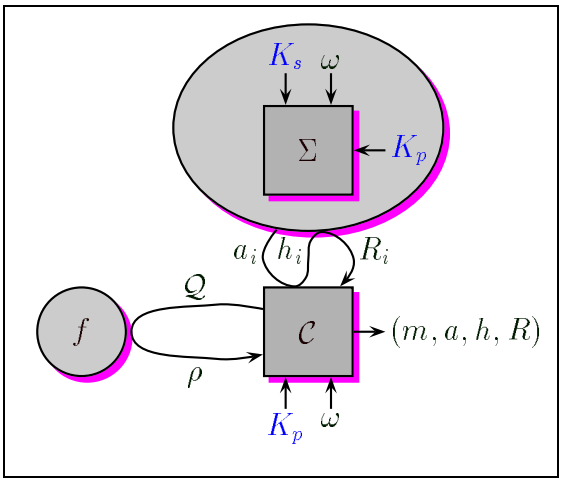

Fig. 6.1 – *Attaques d'un schéma de signature en blanc*

Une falsification possible que l'on veut à tout prix interdire lors d'une utilisation en monnaie électronique est la «falsification supplémentaire».

Définition 84 ($(\ell, \ell+1)$-falsification). Il existe un attaquant $\mathcal{C}$ tel qu'après ℓ interactions avec le signeur, l'attaquant $\mathcal{C}$ parvient à fabriquer $\ell + 1$ signatures valides.

Définition 85 (Falsification supplémentaire). Il existe un entier positif ℓ tel qu'une $(\ell, \ell+1)$-falsification soit possible.

Comme pour la falsification d'un schéma de signature électronique, plusieurs types d'attaques sont imaginables (voir figure 6.1).

– L'attaque séquentielle (voir figure 6.2) : l'attaquant interagit de façon séquentielle avec le signeur. Il prend donc connaissance des premiers tours des interactions (les a_i) les uns après les autres.

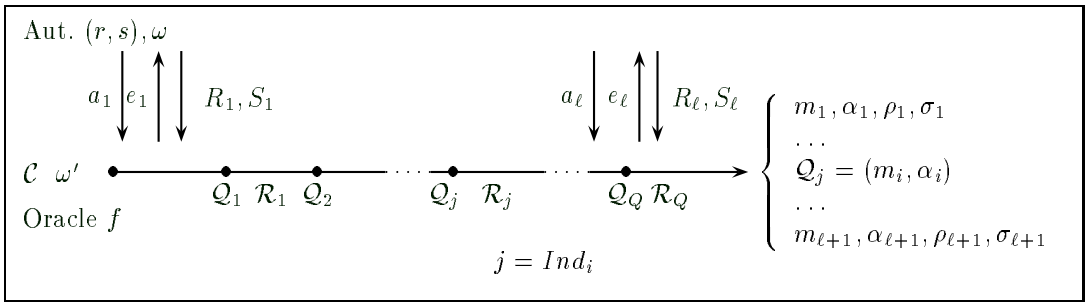

Fig. 6.2 – *Attaque séquentielle d'un schéma de signature en blanc*

– L'attaque parallèle (voir figure 6.3) : l'attaquant peut initier plusieurs interactions avec le signeur et envoyer les «challenges» quand il le souhaite. Mais il a, dès le début, connaissance des premiers tours et peut donc adapter sa stratégie.

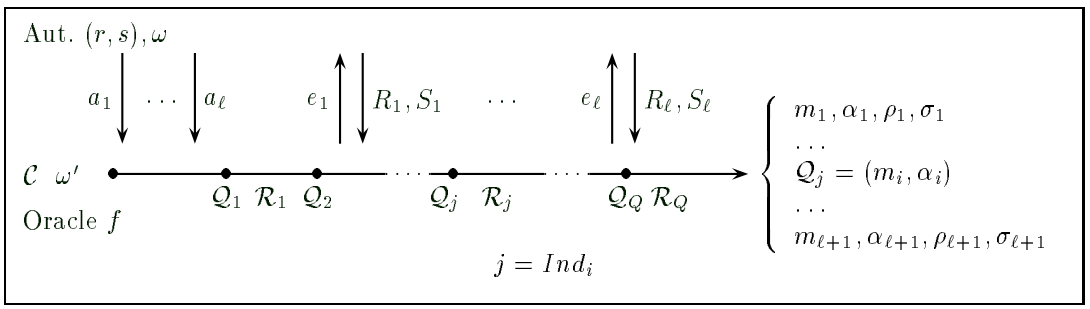

Fig. 6.3 – *Attaque parallèle d'un schéma de signature en blanc*

Il est bien évident que l'attaque parallèle englobe l'attaque séquentielle et est donc plus forte. Cependant, dans le cas de la signature en blanc RSA, qui est simplement en deux passes, ces deux attaques sont équivalentes.

6.3 Premiers schémas avec preuves de sécurité

Dans cette partie, comme dans la partie consacrée aux schémas de signature, nous allons établir un théorème général permettant de prouver la sécurité des schémas de

signature en blanc. Cependant, la propriété «à divulgation nulle de connaissance» du protocole d'identification de base ne suffira plus. En d'autres termes, notre preuve ne conviendra pas pour le schéma de signature de Schnorr en blanc. En effet, on peut aisément voir que la simulation du signeur que nous avons utilisée pour prouver la sécurité du schéma de signature ne convient plus pour simuler le signeur dans le cas de la signature en blanc. Cela provient du fait que nous n'avons plus aucun contrôle sur le «challenge» que le signeur reçoit. Nous avons donc besoin d'un signeur qui possède une clé secrète.

Nous allons donc étudier les signatures en blancs provenant de la transformation de protocoles d'identification à trois passes «à témoin indistinguable» tels que ceux présentés par Okamoto [77].

6.3.1 Schéma générique

Nous allons tout d'abord décrire le format général des schémas qui seront concernés par nos résultats. En effet, nous nous restreindrons aux schémas provenant de protocoles d'identification «à témoin indistinguable» à trois passes. Par conséquent, les signatures obtenues seront des triplets (σ_1, h, σ_2) où $h = f(m, \sigma_1)$, comme dans les schémas précédemment étudiés. La fonction de hachage f sera encore une fois considérée comme un oracle aléatoire fournissant des condensés de longueur $k \gg \log n$, où n est la longueur de la clé publique.

La propriété «à témoin indistinguable» réclamée ici signifie que :

- il existe plusieurs clés secrètes associées à une même clé publique.

- lors de l'interaction avec l'autorité, la vue de l'utilisateur est indépendante de la clé secrète utilisée (le «témoin»).

- la connaissance de deux clés secrètes distinctes associées à une même clé publique permet de résoudre un problème «insoluble» pour tous.

Pour rendre les choses plus concrètes, nous utiliserons dans la suite l'adaptation «à témoin indistinguable» du schéma de Schnorr par Okamoto (voir figure 6.4), et sa transformation en schéma de signature en blanc (voir figure 6.5).

Dans le protocole en question, il est bien évident que deux clés secrètes distinctes associées à un même y, c'est-à-dire $y = g^{-r}h^{-s} = g^{-r'}h^{-s'} \mod p$, fournissent le logarithme discret de h en base g modulo p, ce qui est *a priori* inconnu de tous.

Nous allons maintenant montrer que les schémas de signature en blanc de ce type n'admettent pas de «falsification supplémentaire» suite à un nombre polylogarithmique d'interactions séquentielles ou parallèles avec l'autorité.

Comme pour la sécurité des schémas de signature, nous allons faire tourner deux fois l'attaquant avec deux oracles aléatoires distincts. Nous pourrons alors lui extraire une clé secrète associée à la clé publique de l'autorité. Grâce à la propriété «à témoin indistinguable» du protocole, nous allons montrer que cette clé obtenue est indépendante de celle utilisée par l'autorité.

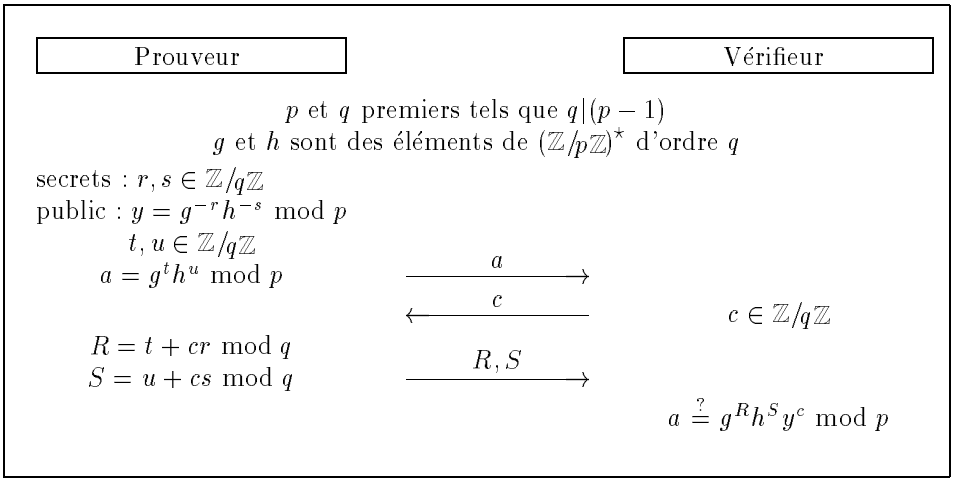

Fig. 6.4 – *Adaptation «à témoin indistinguable» du schéma de Schnorr*

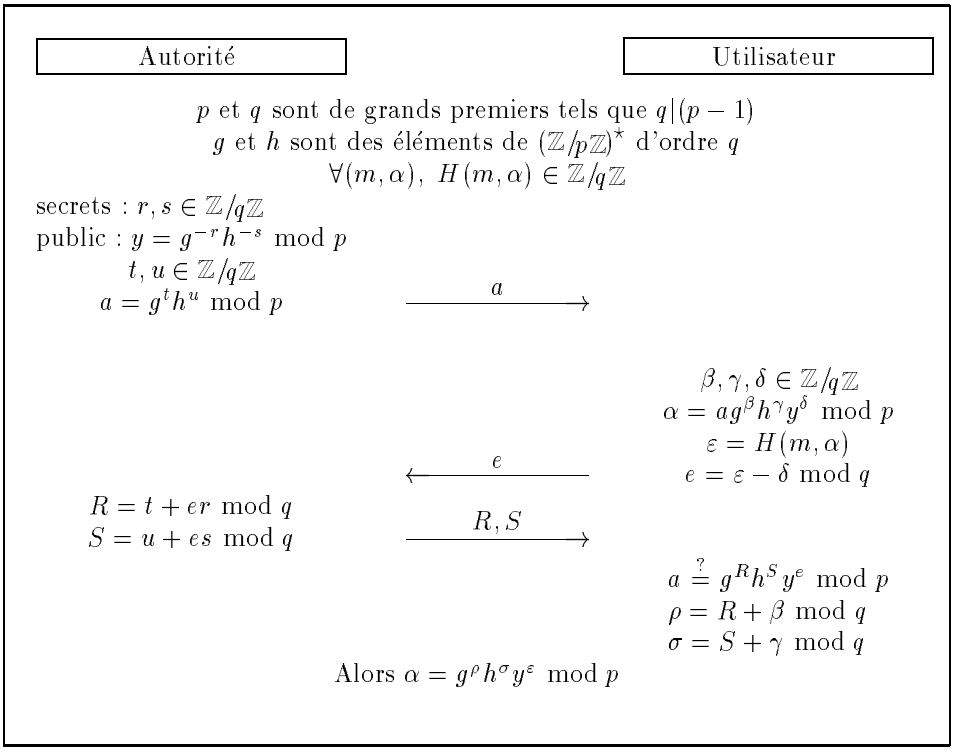

Fig. 6.5 – *Signature en blanc Okamoto–Schnorr*

6.3.2 Preuve de sécurité

6.3.2.1 Énoncé du théorème

Théorème 86 (Sécurité des schémas de signature en blanc). *Considérons le schéma de signature en blanc d'Okamoto–Schnorr dans le modèle de l'oracle aléatoire. Si une «falsification supplémentaire» est possible avec probabilité non-négligeable après un nombre polylogarithmique d'interactions avec le signeur, alors le problème du logarithme discret dans un sous-groupe peut être résolu en temps polynomial.*

6.3.2.2 Preuve

Afin d'établir ce résultat, nous allons donner une idée de la démarche que nous allons suivre, simplifier les notations, puis énoncer un nouveau «lemme de bifurcation» qui achèvera la preuve.

Plan de la preuve Supposons que Charlie, vu comme une machine de Turing Polynomiale Probabiliste à Oracle $\mathcal{C}^f$, soit un attaquant capable de fournir $\ell+1$ signatures, après ℓ interactions avec le signeur et Q questions à l'oracle aléatoire, avec probabilité non-négligeable $\varepsilon \geq 1/P$, où P et Q sont des polynômes en n, la taille de la clé publique, et ℓ un polynôme en $\log n$. Nous noterons (a_i, e_i, R_i, S_i) pour $i \in \{1, \ldots, \ell\}$, les ℓ interactions avec le signeur, $\mathcal{Q}_i$ pour $i \in \{1, \ldots, Q\}$, les Q questions à l'oracle aléatoire, puis $\mathcal{R}_i$ pour $i \in \{1, \ldots, Q\}$, ses réponses. Enfin, Charlie retourne $\ell+1$ signatures valides que nous noterons $(m_i, \alpha_i, \varepsilon_i, \rho_i, \sigma_i)$ pour $i = 1, \ldots, \ell+1$, *i.e.* qui vérifient l'équation caractéristique de la signature. Dans le cas particulier que nous traitons,

$$\alpha_i = g^{\rho_i} h^{\sigma_i} y^{\varepsilon_i} \bmod p \text{ avec } \varepsilon_i = f(m_i, \alpha_i).$$

Les données publiques consistent en deux grands entiers premiers p et q tels que $q \,|\, p-1$, et deux éléments g et h de $(\mathbb{Z}/p\mathbb{Z})^\star$ d'ordre q. L'autorité possède une clé secrète (r, s) stockée sur son «ruban de connaissance».

À l'aide d'une coalition entre l'autorité et l'attaquant, nous voulons résoudre le problème du logarithme discret de h relativement à g modulo p. Nous allons utiliser la technique de l'«oracle replay» comme pour la preuve de sécurité des schémas de signature, formalisée au cours du chapitre précédent et dans [86]. Pour cela, on lance l'attaque avec des clés choisies aléatoirement ainsi que les rubans aléatoires et l'oracle f. On choisit un indice j. On relance l'attaque avec les mêmes clés, les mêmes rubans aléatoires et un nouvel oracle répondant de façon identique aux $j-1$ premières questions. Avec ces deux exécutions, nous espérons obtenir deux écritures distinctes, relativement à g et h, d'un même α_i, celui provenant de la j-ième question, avec probabilité non-négligeable. En effet,

si $\alpha_i = g^{\chi} h^{\varphi} = g^{\chi'} h^{\varphi'} \bmod p$ avec $\chi \neq \chi'$, alors $\log_g h = (\chi - \chi')(\varphi - \varphi')^{-1} \bmod q$. Pour obtenir ce résultat, nous allons également passer par un «lemme de bifurcation» .

Lemme 87 (Lemme de bifurcation).
Si on choisit aléatoirement un indice j et deux séquences de réponses pour l'oracle, identiques jusqu'à l'indice $j-1$. Si, de plus, on choisit aléatoirement la clé secrète (r,s) ainsi que les rubans aléatoires de l'autorité et de l'attaquant. Alors, avec probabilité non-négligeable, l'attaquant fournit deux séries de sorties valides, $(m_i, \alpha_i, \varepsilon_i, \rho_i, \sigma_i)$ et $(m'_i, \alpha'_i, \varepsilon'_i, \rho'_i, \sigma'_i)$ pour $i = 1, \ldots, \ell+1$, telles qu'il existe un indice i vérifiant $\mathcal{Q}_j = (m_i, \alpha_i) = (m'_i, \alpha'_i)$, et donc où les challenges, $\varepsilon_i = \mathcal{R}_j$ et $\varepsilon'_i = \mathcal{R}'_j$, sont deux valeurs distinctes. De plus, ces sorties vérifient

$$\rho_i - r\varepsilon_i \neq \rho'_i - r\varepsilon'_i \bmod q.$$

Simplification des notations Nous allons simplifier les notations pour rendre la démonstration plus lisible. Tout d'abord, de la même manière que pour le *lemme de bifurcation* 65 relatif à la sécurité des schémas de signature, nous pouvons supposer que tous les (m_i, α_i) retournés par l'attaquant étaient des questions posées à l'oracle lors de l'attaque. Dans le cas contraire, la probabilité de succès serait négligeable en raison du caractère aléatoire des sorties de l'oracle aléatoire. Ceci entraîne une faible chute de la probabilité de succès de ε à $3\varepsilon/4$ pour n assez grand (dès que $\varepsilon \geq 4/q$).

Ensuite, pour tout $i \in \{1, \ldots, \ell+1\}$, nous noterons Ind_i l'indice de (m_i, α_i) dans la liste des questions posées à l'oracle aléatoire. Alors $\mathcal{Q}_{Ind_i} = (m_i, \alpha_i)$ et $\mathcal{R}_{Ind_i} = \varepsilon_i$, pour tout i. La coalition autorité-attaquant est modélisée figure 6.6, où le couple (r,s) est la

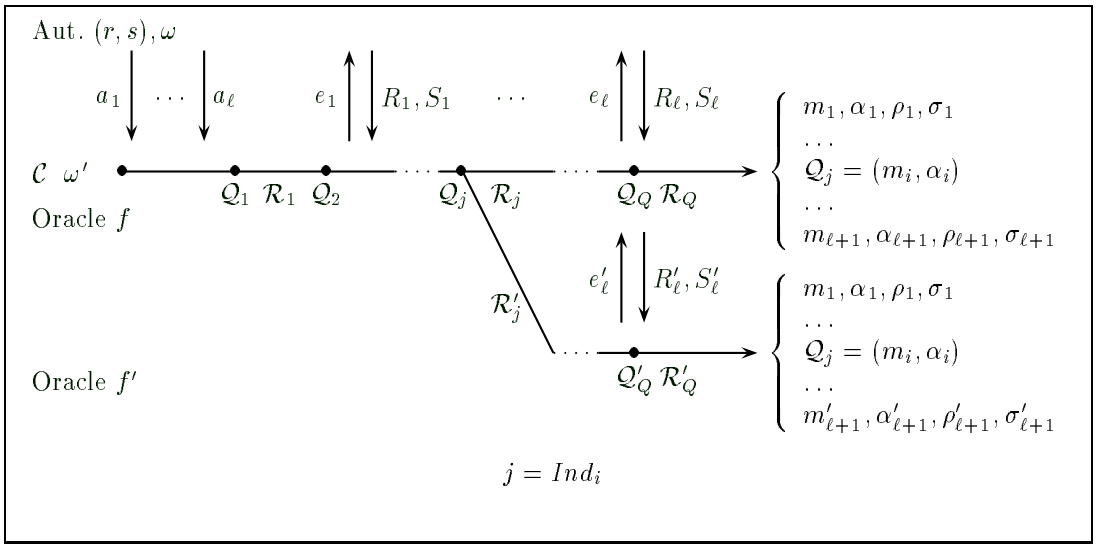

Fig. 6.6 – *Lemme de bifurcation*

clé secrète utilisée par l'autorité, et où le ruban aléatoire ω de l'autorité détermine les couples (t_i, u_i) tels que $a_i = g^{t_i} h^{u_i} \mod p$ pour $i = 1, \ldots, \ell$. La distribution de (r, s, y) où r et s sont tirés aléatoirement et $y = g^{-r} h^{-s} \mod p$ est la même que la distribution de (r, s, y) où r et y d'ordre q sont tirés aléatoirement et s est l'unique élément de $(\mathbb{Z}/q\mathbb{Z})^\star$ tel que $y = g^{-r} h^{-s} \mod p$. Ainsi, nous remplacerons le couple (r, s) par le couple (y, r), et de la même manière, les couples (t_i, u_i) par les couples (a_i, t_i).

Dans la suite, la variable ν regroupera le multiplet $(\omega', y, a_1, \ldots, a_\ell)$. De même, la variable τ regroupera le multiplet $(t_1, \ldots, t_\ell)$, puis on notera $E = (e_1, \ldots, e_\ell)$.

Nous noterons alors $\mathcal{S}$ l'ensemble des multiples (ν, r, τ, f) qui mènent à un succès tel que toutes les mises en gages (m_i, α_i) des signatures fabriquées ont été posées à l'oracle aléatoire lors de l'attaque.

$$\Pr_{\nu,r,\tau,f}[(\nu, r, \tau, f) \in \mathcal{S}] \geq \frac{3\varepsilon}{4}.$$

Preuve Nous voulons maintenant prouver qu'après un rejeu, avec probabilité non négligeable, nous obtenons une sortie commune α_i telle que

$$\begin{array}{rcl} \alpha_i = g^{\rho_i} h^{\sigma_i} y^{\varepsilon_i} & = & g^{\rho_i - r\varepsilon_i} h^{\sigma_i - s\varepsilon_i} \mod p \\ = g^{\rho'_i} h^{\sigma'_i} y^{\varepsilon'_i} & = & g^{\rho'_i - r\varepsilon'_i} h^{\sigma'_i - s\varepsilon'_i} \mod p \end{array} \quad \text{avec } \rho_i - r\varepsilon_i \neq \rho'_i - r\varepsilon'_i \mod q.$$

Nous sommes alors conduits à étudier, pour tout $i \in \{1, \ldots, \ell + 1\}$, la variable aléatoire $\chi_i = \rho_i - r\varepsilon_i \mod q$. Nous pouvons tout d'abord remarquer que, pour tout i, α_i ne dépend que de ν, r, τ et de f_{Ind_i}, la restriction de f aux $Ind_i - 1$ premières réponses. La question fondamentale est de savoir si la variable aléatoire $\chi_i = \rho_i - r\varepsilon_i \mod q$ est sensible ou non aux réponses fournies par l'oracle aux rangs Ind_i, $Ind_i + 1$, etc. Nous espérons une réponse par l'affirmative. Une manière d'aborder cette question est d'étudier la valeur de plus grande probabilité prise par cette variable quand (ν, r, τ) et f_j sont fixées.

Nous allons pour cela considérer les fonctions $c_{i,j}(\nu, r, \tau, f_j)$, pour $i = 1, \ldots, \ell + 1$ et $j = 1, \ldots, Q$, où f_j prend des valeurs dans l'ensemble des réponses possibles aux $j - 1$ premières questions. Pour cela, posons

$$\lambda_{i,j}(\nu, r, \tau, f_j, c) = \Pr_f \left[\begin{array}{c} (\nu, r, \tau, f) \in \mathcal{S} \ \wedge \ Ind_i = j \\ \wedge \ \chi_i(\nu, r, \tau, f) = c \end{array} \middle| f \text{ prolonge } f_j \right].$$

Nous définissons alors la fonction $c_{i,j}(\nu, r, \tau, f_j)$ par une valeur c qui maximise la fonction $\lambda_{i,j}(\nu, r, \tau, f_j, c)$. Nous pouvons désormais définir le sous-ensemble $\mathcal{G}$ de $\mathcal{S}$ dont les éléments satisfont, pour tout i, $\chi_i(\nu, r, \tau, f) = c_{i,j}(\nu, r, \tau, f_j)$, où $j = Ind_i(\nu, r, \tau, f)$. Nous noterons $\mathcal{B}$ son complémentaire dans $\mathcal{S}$.

Définition 88. Nous noterons Φ la transformation qui à tout quadruplet (ν, r, τ, f) associe $(\nu, r + 1, \tau - E, f)$, où $\tau - E = (t_1 - e_1, \ldots, t_\ell - e_\ell)$.

Cette transformation possède d'importantes propriétés (voir figure 6.7).

Lemme 89 (Bijectivité).
Les deux exécutions correspondant à (ν, r, τ, f) et $\Phi(\nu, r, \tau, f)$ sont parfaitement identiques pour l'attaquant En particulier, les sorties sont les mêmes.

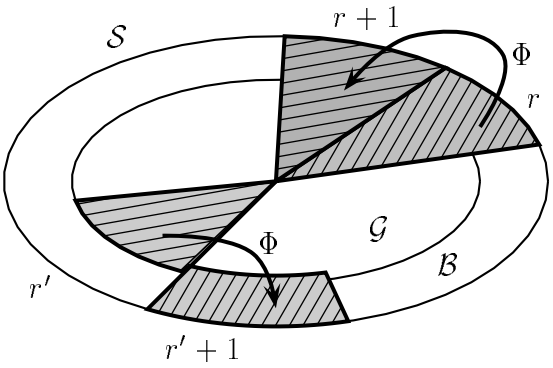

Fig. 6.7 – *Propriétés de* Φ

Preuve. Soit (ν, r, τ, f) une entrée de la coalition. Si on relance l'attaque avec $r' = r + 1$ et $\tau' = \tau - E$, mais le même ν et le même oracle f, alors les réponses de l'oracle f sont inchangées et les interactions avec l'autorité deviennent

$$R'_i(r', t'_i, e_i) = t'_i + r'e_i = (t_i - e_i) + (r+1)e_i = t_i + re_i = R_i(r, t_i, e_i).$$

Par conséquent, le tout reste inchangé. □

Corollaire 90 .
Φ est une bijection de $\mathcal{S}$ dans $\mathcal{S}$.

Le lemme suivant montre que la transformation Φ envoie tout le sous-ensemble $\mathcal{G}$ dans le sous-ensemble $\mathcal{B}$, à part une fraction négligeable.

Lemme 91 (Unicité).
Pour un triplet (ν, r, τ) fixé,

$$\Pr_f[((\nu, r, \tau, f) \in \mathcal{G}) \wedge (\Phi(\nu, r, \tau, f) \in \mathcal{G})] \leq Q^{\ell+1}/q.$$

Preuve. Montrons ceci par l'absurde. Supposons qu'il existe un triplet (ν, r, τ) tel que

$$\Pr_f[((\nu, r, \tau, f) \in \mathcal{G}) \wedge (\Phi(\nu, r, \tau, f) \in \mathcal{G})] > Q^{\ell+1}/q.$$

Alors il existe un $\ell + 1$-uplet $(u_1, \ldots, u_{\ell+1})$ tel que

$$\Pr_f[((\nu, r, \tau, f) \in \mathcal{G}) \wedge (\Phi(\nu, r, \tau, f) \in \mathcal{G}) \wedge (\forall i, Ind_i = u_i)] > 1/q.$$

De même, il existe un ℓ-uplet $(\tilde{e}_1, \ldots, \tilde{e}_\ell)$ tel que

$$\Pr_f\left[\begin{array}{c}((\nu, r, \tau, f) \in \mathcal{G}) \wedge (\Phi(\nu, r, \tau, f) \in \mathcal{G}) \\ \wedge(\forall i \in \{1, \ldots, \ell+1\}, Ind_i = u_i) \wedge (\forall i \in \{1, \ldots, \ell\}, e_i = \tilde{e}_i)\end{array}\right] > 1/q^{\ell+1}.$$

Par suite, il existe deux oracles aléatoires f et f' qui fournissent des réponses distinctes uniquement à certaines questions $Q_{u_i} : \varepsilon_i \neq \varepsilon'_i$. Nous noterons i le plus petit de ces indices et j l'indice de la question à l'oracle associée ($j = u_i$). Alors $f_j = f'_j$. De plus, f et f' sont tels que $(\nu, r, \tau, f) \in \mathcal{G}$, $\Phi(\nu, r, \tau, f) \in \mathcal{G}$ et $(\nu, r, \tau, f') \in \mathcal{G}$, $\Phi(\nu, r, \tau, f') \in \mathcal{G}$. Ainsi, par définition de $\mathcal{G}$,

$$c_{i,j}(\nu, r, \tau, f_j) = \rho_i(\nu, r, \tau, f) - r\varepsilon_i = \rho_i(\Phi(\nu, r, \tau, f)) - r\varepsilon_i = c_{i,j}(\nu, r+1, \tau\text{-}E, f_j) + \varepsilon_i,$$
$$c_{i,j}(\nu, r, \tau, f'_j) = \rho_i(\nu, r, \tau, f') - r\varepsilon'_i = \rho_i(\Phi(\nu, r, \tau, f')) - r\varepsilon'_i = c_{i,j}(\nu, r+1, \tau\text{-}E', f'_j) + \varepsilon'_i.$$

L'égalité entre les oracles partiels $f_j = f'_j$ entraîne $c_{i,j}(\nu, r, \tau, f_j) = c_{i,j}(\nu, r, \tau, f'_j)$. De plus, on a supposé $E = E' = (\tilde{e}_1, \ldots, \tilde{e}_\ell)$, donc $c_{i,j}(\nu, r+1, \tau - E, f_j) = c_{i,j}(\nu, r+1, \tau - E', f'_j)$. Par suite, $\varepsilon_i = \varepsilon'_i$, ce qui contredit l'hypothèse. On en déduit

$$\Pr_f[((\nu, r, \tau, f) \in \mathcal{G}) \wedge (\Phi(\nu, r, \tau, f) \in \mathcal{G})] \leq \frac{Q^{\ell+1}}{q}.$$

□

Ainsi, nous pouvons dire que

$$\begin{aligned}
\Pr[\mathcal{G}] &= \Pr[((\nu, r, \tau, f) \in \mathcal{G}) \wedge (\Phi(\nu, r, \tau, f) \in \mathcal{G})] \\
&+ \Pr[((\nu, r, \tau, f) \in \mathcal{G}) \wedge (\Phi(\nu, r, \tau, f) \in \mathcal{B})] \\
&\leq \frac{Q^{\ell+1}}{q} + \Pr[\Phi(\nu, r, \tau, f) \in \mathcal{B}] \leq \frac{Q^{\ell+1}}{q} + \Pr[\mathcal{B}].
\end{aligned}$$

Alors, $\Pr[\mathcal{B}] \geq 1/2 \cdot (\Pr[\mathcal{S}] - Q^{\ell+1}/q) \geq (3\varepsilon/4 - Q^{\ell+1}/q)/2$. Pour n assez grand, nous avons $\varepsilon \geq 4Q^{\ell+1}/q$. Dans ce cas, $\Pr[\mathcal{B}] \geq \varepsilon/4$.

Conclusion Nous savons que

$$\begin{aligned}
&\sum_{i=1}^{\ell+1} \Pr_{\nu,r,\tau,f}[(\chi_i(\nu, r, \tau, f) \neq c_{i,Ind_i}(\nu, r, \tau, f_{Ind_i})) \wedge \mathcal{S}] \\
&\geq \Pr_{\nu,r,\tau,f}[((\exists i)\ \chi_i(\nu, r, \tau, f) \neq c_{i,Ind_i}(\nu, r, \tau, f_{Ind_i})) \wedge \mathcal{S}] \\
&= \Pr[\mathcal{B}] \geq \frac{\varepsilon}{4}.
\end{aligned}$$

Alors il existe $i \in \{1, \ldots, \ell+1\}$ tel que

$$\Pr[(\chi_i(\nu, r, \tau, f) \neq c_{i,Ind_i}(\nu, r, \tau, f_{Ind_i})) \wedge \mathcal{S}] \geq \frac{\varepsilon}{4(\ell+1)}.$$

De la même manière, il existe $j \in \{1, \ldots, Q\}$ tel que

$$\Pr[(\chi_i(\nu, r, \tau, f) \neq c_{i,j}(\nu, r, \tau, f_j)) \wedge \mathcal{S} \wedge (Ind_i = j)] \geq \frac{\varepsilon}{4(\ell+1)Q}.$$

Le lemme de séparation 28 garantit l'existence d'un ensemble X tel que

i) $\Pr[(\nu, r, \tau, f_j) \in X] \geq \dfrac{\varepsilon}{8(\ell+1)Q}$

ii) pour tout $(\nu, r, \tau, f_j) \in X$,
$$\Pr_f\left[\begin{array}{c} \mathcal{S} \wedge Ind_i = j \\ \wedge\ (\chi_i(\nu, r, \tau, f) \neq c_{i,j}(\nu, r, \tau, f_j)) \end{array} \middle| f \text{ prolonge } f_j\right] \geq \frac{\varepsilon}{8(\ell+1)Q}.$$

Choisissons alors aléatoirement un couple (i, j). Ainsi, avec probabilité supérieure à $1/(\ell+1)Q$, nous avons fait un bon choix. Choisissons maintenant un quadruplet aléatoire (ν, r, τ, f). Avec probabilité supérieure à $(\varepsilon/8(\ell+1)Q)^2$, $(\nu, r, \tau, f_j) \in X$, $(\nu, r, \tau, f) \in \mathcal{S}$, $Ind_i = j$ et $\chi_i(\nu, r, \tau, f) \neq c_{i,j}(\nu, r, \tau, f_j)$.

Nous noterons d la valeur $\chi_i(\nu, r, \tau, f)$ et c la valeur $c_{i,j}(\nu, r, \tau, f_j)$.

Deux cas se présentent alors :

1. $\lambda_{i,j}(\nu, r, \tau, f_j, d) \geq \varepsilon/16(\ell+1)Q$, donc, par définition des fonctions $c_{i,j}$,
$$\lambda_{i,j}(\nu, r, \tau, f_j, c) \geq \varepsilon/16(\ell+1)Q.$$
2. Dans le cas contraire,
$$\begin{aligned}
&\lambda_{i,j}(\nu, r, \tau, f_j, d) + \Pr_{f'}\left[\begin{array}{c}\mathcal{S} \wedge Ind_i = j \\ \wedge\Big(\chi_i(\nu, r, \tau, f') \neq d\Big)\end{array} \;\middle|\; f' \text{ prolonge } f_j\right] \\
&= \Pr_{f'}\left[\mathcal{S} \wedge Ind_i = j \mid f' \text{ prolonge } f_j\right] \\
&\geq \Pr_{f'}\left[\begin{array}{c}\mathcal{S} \wedge Ind_i = j \\ \wedge\Big(\chi_i(\nu, r, \tau, f') \neq c\Big)\end{array} \;\middle|\; f' \text{ prolonge } f_j\right] \;\geq\; \varepsilon/8(\ell+1)Q.
\end{aligned}$$

Les deux cas mènent à

$$\Pr_{f'}\left[\begin{array}{c}\mathcal{S} \wedge Ind_i = j \\ \wedge\Big(\chi_i(\nu, r, \tau, f') \neq d\Big)\end{array} \;\middle|\; f' \text{ prolonge } f_j\right] \geq \varepsilon/16(\ell+1)Q.$$

Complexité globale de la réduction En utilisant la technique du «oracle replay» avec un index de bifurcation aléatoire, la probabilité de succès est supérieure à

$$\frac{1}{(\ell+1)Q} \times \left(\frac{\varepsilon}{8(\ell+1)Q}\right)^2 \times \frac{\varepsilon}{16Q(\ell+1)} = \frac{\varepsilon^3}{1024(\ell+1)^4Q^4} \geq \frac{\varepsilon^3}{16384\ell^4Q^4}.$$

où Q est le nombre de questions posées à l'oracle aléatoire et ε la probabilité de succès d'une $(\ell, \ell+1)$-falsification, supposée supérieure à $4Q^{\ell+1}/q$.

6.3.3 Réduction plus performante

Plus précisément, nous venons de prouver le résultat suivant :

Théorème 92 (Lemme de bifurcation).
Supposons que la machine de Turing $\mathcal{C}$ soit capable d'effectuer une falsification supplémentaire en temps T, avec probabilité $\varepsilon \geq 1/P$, après Q appels à l'oracle aléatoire et ℓ appels à l'autorité. Si de plus, $\varepsilon \geq 4Q^{\ell+1}/q$, alors il existe une machine de Turing $\mathcal{M}$ capable de fournir deux écritures distinctes d'un même élément, ***en temps $T' = 2T$ et avec probabilité $\varepsilon' \geq \varepsilon^3/16384\ell^4Q^4$.***

Mais une machine plus performante peut être construite en utilisant le lemme de séparation 28 conjugué au lemme de succès 29.

Théorème 93 (Lemme de bifurcation amélioré).
Supposons que la machine de Turing $\mathcal{C}$ soit capable d'effectuer une falsification supplémentaire en temps T, avec probabilité $\varepsilon \geq 1/P$, après Q appels à l'oracle aléatoire et ℓ appels à l'autorité. Si de plus, $\varepsilon \geq 4Q^{\ell+1}/q$, alors il existe une machine de Turing $\mathcal{M}'$ capable de fournir 2 écritures distinctes d'un même élément, ***en temps $T' \leq 33Q\ell T/\varepsilon$ et avec probabilité $\varepsilon' \geq 1/72\ell^2$.***

Preuve. Supposons la probabilité de succès de $\mathcal{C}$ supérieure ou égale à ε. Voici la stratégie de $\mathcal{M}'$:

- $\mathcal{M}'$ choisit un indice $i \in \{1, \ldots, \ell+1\}$.
- Première étape :
  1. $\mathcal{M}'$ initialise le compteur $c = 0$.
  2. $\mathcal{M}'$ choisit la clé secrète (r, s), calcule y, et choisit deux rubans aléatoires ω et ω' ainsi qu'un oracle aléatoire f.
  3. $\mathcal{M}'$ fait tourner $\mathcal{C}(\nu, r, \tau, f)$.
  4. Si elle n'obtient pas une falsification supplémentaire avec un Ind_i fini et si $c < 1/\varepsilon$, alors elle incrémente c et retourne en 2.
  5. Si $c > 1/\varepsilon$, $\mathcal{M}'$ retourne `Échec`, sinon, elle note
     - $(m_i, \alpha_i, \varepsilon_i, \rho_i, \sigma_i)$ la i-ième signature obtenue,
     - j la valeur de Ind_i,
     - f_j la restriction de f aux $j-1$ premières valeurs,
     - $d = \chi_i$.
- Deuxième étape :
  1. $\mathcal{M}'$ réinitialise le compteur $c = 0$.
  2. $\mathcal{M}'$ choisit un deuxième prolongement f' de f_j.
  3. $\mathcal{M}'$ fait tourner $\mathcal{C}(\nu, r, \tau, f')$ et incrémente c.
  4. Si elle n'obtient pas une falsification supplémentaire avec $Ind_i = j$ et $\chi_i \neq d$, et si $c < 16Q(\ell+1)/\varepsilon$, alors elle retourne en 2.
  5. Si $c > 16Q(\ell+1)/\varepsilon$, $\mathcal{M}'$ retourne `Échec`, sinon, elle note $(m_i, \alpha_i, \varepsilon'_i, \rho'_i, \sigma'_i)$ la i-ième signature obtenue.
- $\mathcal{M}'$ retourne les deux couples $(\rho_i - r\varepsilon_i, \sigma_i - s\varepsilon_i)$ et $(\rho'_i - r\varepsilon'_i, \sigma'_i - s\varepsilon'_i)$.

Comme on l'a déjà vu, avec probabilité supérieure à $1/(\ell+1)$, l'indice i choisi satisfait

$$\Pr[(\chi_i(\nu,r,\tau,f) \neq c_{i,Ind_i}(\nu,r,\tau,f_{Ind_i})) \,|\, \mathcal{S}] \geq \frac{1}{4(\ell+1)}.$$

Or, avec probabilité supérieure à $1/3$, la première étape nous fournit un succès. Dans ce cas, avec probabilité supérieure à $1/4(\ell+1)$, $\chi_i \neq c_{i,Ind_i}$. Définissons

$$E = \left\{ (j,\nu,r,\tau,f_j) \middle| \Pr_f[\mathcal{S} \wedge (\chi_i \neq c_{i,j}) \wedge (Ind_i = j) \,|\, f \text{ prolonge } f_j] \geq \frac{\varepsilon}{8Q(\ell+1)} \right\}.$$

Posons $j = Ind_i$. Quelle est alors la probabilité pour que le succès obtenu satisfasse $(j,\nu,r,\tau,f_j) \in E$?

$$\begin{aligned}
\Pr[E \,|\, \mathcal{S} \wedge \chi_i \neq c_{i,Ind_i}] &= \Pr[E \wedge \mathcal{S} \wedge \chi_i \neq c_{i,Ind_i}] / \Pr[\mathcal{S} \wedge \chi_i \neq c_{i,Ind_i}] \\
&= 1 - \left(\Pr[\mathcal{S} \wedge \chi_i \neq c_{i,Ind_i} \wedge \bar{E}] / \Pr[\mathcal{S} \wedge \chi_i \neq c_{i,Ind_i}] \right) \\
&= 1 - \Pr[\mathcal{S} \wedge \chi_i \neq c_{i,Ind_i} \,|\, \bar{E}] \Pr[\bar{E}] / \Pr[\mathcal{S} \wedge \chi_i \neq c_{i,Ind_i}] \\
&\geq 1 - \frac{4(\ell+1)}{\varepsilon} \Pr[\mathcal{S} \wedge \chi_i \neq c_{i,Ind_i} \,|\, \bar{E}] \\
&\geq 1 - \frac{4(\ell+1)}{\varepsilon} \sum_j \Pr[\mathcal{S} \wedge Ind_i = j \wedge \chi_i \neq c_{i,j} \,|\, \bar{E}] \\
&\geq 1 - \frac{4(\ell+1)}{\varepsilon} \sum_j \frac{\varepsilon}{8Q(\ell+1)} \;\geq\; \frac{1}{2}.
\end{aligned}$$

Ainsi, globalement, avec probabilité supérieure à $1/3$, nous obtenons un succès au cours de la première étape. Et avec probabilité supérieure à $1/8(\ell+1)$, ce succès vérifie $\chi_i \neq c_{i,j}$ et $(j,\nu,r,\tau,f_j) \in E$, en posant $j = Ind_i$.

Comme nous l'avons déjà vu, lors de la deuxième étape,

$$\Pr_{f'}\left[\mathcal{S} \wedge Ind_i = j \wedge \Big(\chi_i(\nu,r,\tau,f') \neq d\Big) \middle| f' \text{ prolonge } f_j \right] \geq \varepsilon/16(\ell+1)Q.$$

Alors nous obtenons un deuxième succès exploitable avec probabilité supérieure à $1/3$. D'où la complexité totale. □

Commentaire Cette fois-ci, la contrainte la plus limitante est celle imposée à ε. En effet, notre preuve réclame $\varepsilon \geq 4Q^{\ell+1}/q$. Elle est également due au fait que lors des deux attaques, la liste des «indices» doit être la même. Cette restriction limite donc la preuve aux attaques polylogarithmiques.

6.4 Applications

Nous allons appliquer ce résultat de sécurité à quelques nouveaux schémas de signature en blanc. Tout d'abord, nous allons énoncer les théorèmes pour les transformations des

deux adaptations d'Okamoto [77]. Ensuite, nous allons démontrer un résultat semblable pour un tout nouveau schéma équivalent à la factorisation dérivé du schéma de Fiat-Shamir [41]. Nous terminerons par un dernier schéma qui ne semble sûr que face à des attaques séquentielles.

6.4.1 Protocoles de Okamoto

Ces résultats s'appliquent tout d'abord aux deux variantes «à témoin indistinguable» d'Okamoto [77]. Le premier a déjà été présenté figure 6.4 et est une variante du schéma d'identification de Schnorr [98, 99]. Sa transformation en schéma de signature en blanc (figure 6.5) a servi d'exemple pour la preuve. Le précédent théorème 86 prouve qu'une «falsification supplémentaire» après un nombre polylogarithmique d'interactions avec le signeur entraîne l'existence d'une machine de Turing Polynomiale Probabiliste capable de fournir une deuxième clé secrète (r', s') distincte de (r, s), avec probabilité non-négligeable, et ce pour une infinité de tailles n, et pour une fraction non-négligeable de données publiques g et h et de clés publiques $y = g^{-r}h^{-s} \bmod p$. On peut alors déterminer le logarithme de h en base g. Ceci prouve qu'une «falsification supplémentaire» de ce schéma est équivalente au cassage du logarithme discret dans les sous-groupes pour une fraction non-négligeable de couples (g, h) avec probabilité non-négligeable. Avec la même extension que celle utilisée pour la preuve du théorème 75 pour la sécurité de El Gamal, nous pouvons énoncer le théorème suivant :

Théorème 94 .
Dans le modèle de l'oracle aléatoire, une «falsification supplémentaire» du schéma de signature en blanc Okamoto–Schnorr selon une «attaque parallèle polylogarithmique» est équivalente au logarithme discret dans un sous-groupe.

Le deuxième schéma proposé par Okamoto est une variante du schéma d'identification de Guillou-Quisquater [51, 52], et est alors équivalent au problème RSA. Le schéma d'identification «à témoin indistinguable» est présenté figure 6.8. Sa transformation en schéma de signature en blanc est présentée figure 6.9.

En appliquant le théorème 86 à ce schéma, on obtient qu'une «falsification supplémentaire» après un nombre polylogarithmique d'interactions parallèles avec le signeur entraîne l'existence d'une machine de Turing Polynomiale Probabiliste capable de fournir une deuxième clé secrète (r', s') distincte de (r, s), avec probabilité non-négligeable, et ce pour une infinité de tailles n, et pour une fraction non-négligeable de données publiques a et λ ainsi que de clés publiques $v = a^{-r}s^{-\lambda} \bmod N$.
Avec $y = a^{-r}s^{-\lambda} = a^{-r'}s'^{-\lambda} \bmod N$, on obtient $a^{r-r'} = (s'/s)^{\lambda} \bmod N$. Comme $r - r'$ est premier avec λ, l'égalité de Bezout fournit deux entiers u et v tels que $u(r - r') + v\lambda = 1$. Cette égalité entraîne $a^{1-v\lambda} = (s'/s)^{u\lambda} \bmod N$. Ceci fournit une racine λ-ième de a,

$$a = \left((s'/s)^u a^v\right)^{\lambda} \bmod N.$$

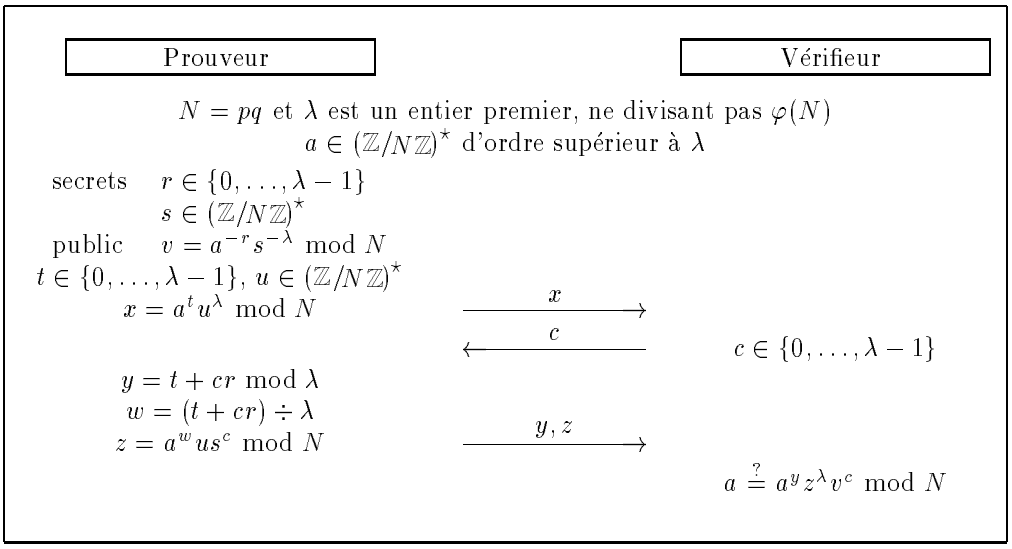

Fig. 6.8 – *Adaptation «à témoin indistinguable» du schéma de Guillou-Quisquater*

Une extension similaire à celle utilisée pour la preuve du théorème 75 pour la sécurité de El Gamal nous permet d'énoncer le théorème suivant :

Théorème 95 .
Dans le modèle de l'oracle aléatoire, une «falsification supplémentaire» du schéma de signature en blanc Okamoto–Guillou-Quisquater, selon une «attaque parallèle polylogarithmique», est équivalente au problème RSA.

6.4.2 Schéma équivalent à la factorisation

Dans cette partie, nous allons proposer un nouveau schéma de signature en blanc équivalent à la factorisation. Ce schéma (voir figure 6.10) est un dérivé du schéma d'identification de Fiat-Shamir à plusieurs secrets avec un module de Blum qui est, comme les variantes d'Okamoto, à témoin indistinguable.

Un module de Blum est un module RSA $N = pq$ tel que $p = 3 \bmod 4$ et $q = 3 \bmod 4$. Dans ce cas, chaque résidu quadratique de $(\mathbb{Z}/N\mathbb{Z})^\star$ admet exactement 4 racines, chacune dans une classe de résiduosité distincte.

On note T l'unique racine de l'unité modulo N comprise entre 2 et $N/2$. Les quatre racines de l'unité sont donc 1, T, -1 et $-T$ respectivement égales à T^0, T^1, T^2, T^3 et qui représentent chacunes une des quatre classes de résiduosité modulo N. Puis, on définit la fonction de résiduosité $C(x)$ par l'unique i tel que x soit dans la même classe de résiduosité que T^i, c'est-à-dire tel que $x/T^{C(x)}$ est un résidu quadratique. On note enfin $\eta(x) = C(x) \bmod 2 \in \{0, 1\}$, la fonction de résiduosité binaire. Il est bon de remarquer dès maintenant que cette fonction de résiduosité binaire est un morphisme de groupe de $(\mathbb{Z}/N\mathbb{Z})^\star$ dans $\{0, 1\}$. En effet, $\eta(1) = 0$ et pour tout couple (x, y) d'éléments de $(\mathbb{Z}/N\mathbb{Z})^\star$,

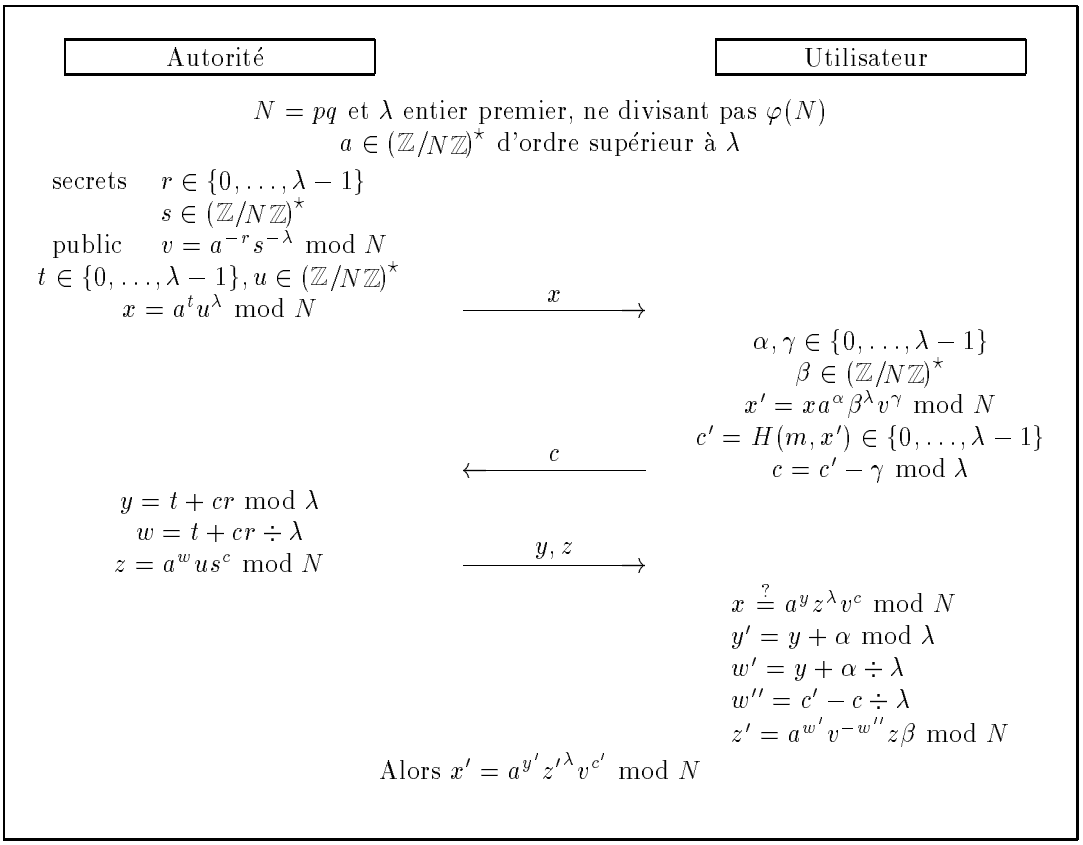

Fig. 6.9 – *Signature en blanc Okamoto–Guillou-Quisquater*

$\eta(xy) = \eta(x) \oplus \eta(y)$. De plus, la connaissance de deux éléments x et y de $(\mathbb{Z}/N\mathbb{Z})^\star$ de même carré mais de résiduosités binaires différentes permet la factorisation de N. On rappellera que $\odot$ représente le produit scalaire de deux vecteurs binaires modulo 2. Pour la validité de ce protocole, il suffit de remarquer que, pour tout couple de bits (b, c), $b \oplus c = \mid b - c \mid$.

En temps polynomial et sur une fraction non-négligeable de clés publiques, l'application du *lemme de bifurcation* 87 à une «falsification supplémentaire» de ce schéma fournit deux écritures «utiles»

$$\begin{aligned} \alpha &= \rho^2 \prod_m V_m{}^{-\varepsilon_m} = \rho'^2 \prod_m V_m{}^{-\varepsilon'_m} \\ &= \left(\rho \prod_m S_m{}^{-\varepsilon_m}\right)^2 = \left(\rho' \prod_m S_m{}^{-\varepsilon'_m}\right)^2 \bmod N, \end{aligned}$$

avec probabilité non-négligeable. Par écritures «utiles», on entend

$$\eta\left(\rho \prod_m S_m{}^{-\varepsilon_m}\right) \neq \eta\left(\rho' \prod_m S_m{}^{-\varepsilon'_m}\right).$$

C'est-à-dire, $\eta(\rho) \oplus (\varepsilon \odot r) \neq \eta(\rho') \oplus (\varepsilon' \odot r)$, où r (resp. ε et ε') est le vecteur ayant pour

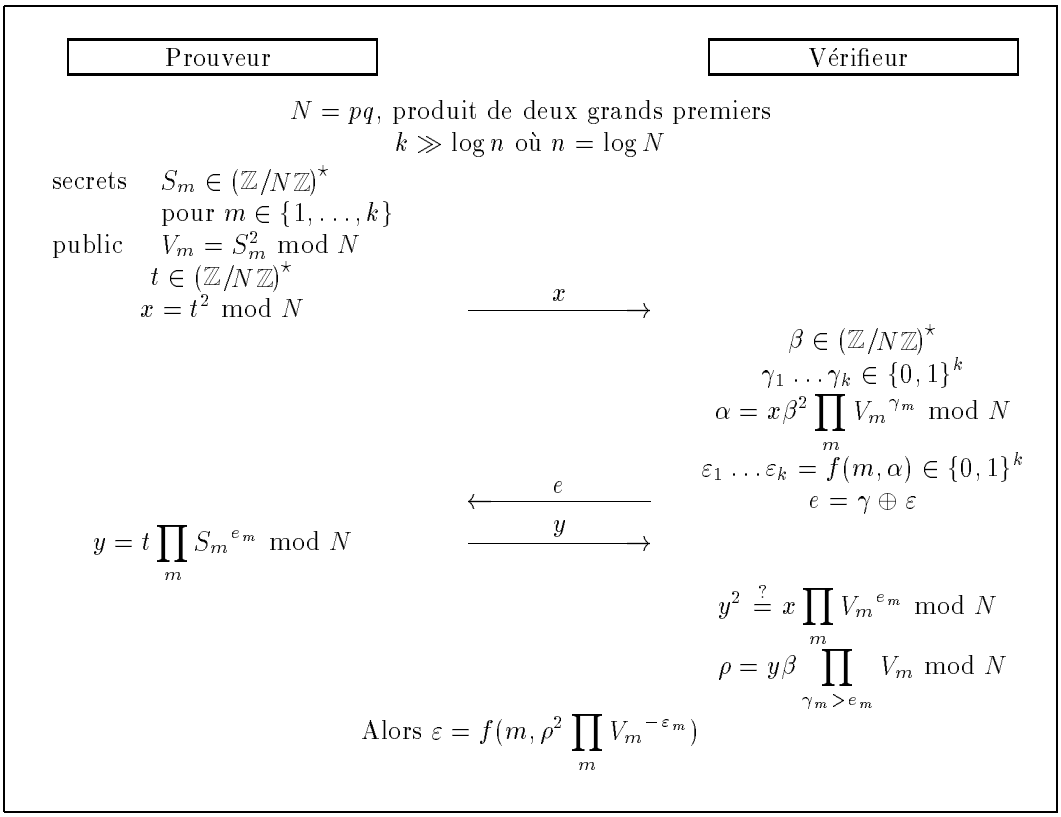

Fig. 6.10 – *Signature en blanc Fiat-Shamir*

composantes les $r_i = \eta(S_i)$ (resp. ε_i et ε'_i). En effet, cette situation fournit un facteur non trivial de N.

On peut alors énoncer le théorème suivant :

Théorème 96 .
Dans le modèle de l'oracle aléatoire, une «falsification supplémentaire» du schéma de signature en blanc dérivé du schéma de Fiat-Shamir à plusieurs secrets selon une «attaque parallèle polylogarithmique» est équivalente à la factorisation des modules de Blum.

Preuve. Comme pour la preuve de sécurité du schéma de signature en blanc d'Okamoto–Schnorr, on considère un attaquant $\mathcal{C}$ capable de fournir $\ell + 1$ signatures après simplement ℓ interactions avec le signeur et Q questions à l'oracle aléatoire, avec probabilité non-négligeable $\varepsilon \geq 1/P$, où P et Q sont des polynômes en n et ℓ un polynôme en $\log n$. Nous noterons (x_i, e_i, y_i) pour $i = 1, \ldots, \ell$, les ℓ interactions avec le signeur, $\mathcal{Q}_i$ pour $i = 1, \ldots, Q$, les Q questions à l'oracle aléatoire, puis $\mathcal{R}_i$ ses réponses. Enfin, la machine $\mathcal{C}$ retourne $\ell + 1$ signatures valides que nous noterons $(m_i, \alpha_i, \varepsilon_i, \rho_i)$ pour $i = 1, \ldots, \ell + 1$. Ces signatures satisfont $\alpha_i = {\rho_i}^2 \prod_m {V_m}^{-\varepsilon_{i,m}} \bmod N$, avec $\varepsilon_{i,1} \ldots \varepsilon_{i,k} = f(m_i, \alpha_i)$.

À chaque clé publique $(V_m)_{1\leq m\leq k}$ sont associées 2^k clés secrètes distinctes, si on ne considère qu'une seule racine par classe de résiduosité binaire :

$$\text{pour } 1 \leq m \leq k \quad \left\{ \begin{array}{l} S_m = T^{r_m}\Sigma_m \bmod N, \\ \text{où } \Sigma_m \text{ est la seule racine résidu quadratique de } V_m. \end{array} \right.$$

pour $r = r_1 \dots r_k \in \{0,1\}^k$.

On regroupera ω, le ruban aléatoire de $\mathcal{C}$, V_1, ..., V_k et x_1, ..., x_k sous la variable ν. Quant à la variable τ, elle regroupera le multiplet $(t_1, \dots, t_k)$. Une exécution sera parfaitement déterminée par le quadruplet (ν, r, τ, f).

On définit alors l'ensemble $\mathcal{S}$ de ces quadruplets qui mènent à un succès tel que toutes les questions (m_i, α_i) ont été posées à l'oracle pendant l'attaque. Pour n assez grand, $\Pr[\mathcal{S}] \geq 3\varepsilon/4$.

On veut, après rejeu, obtenir deux racines carrées d'un même α_i,

$$\alpha_i = \left(\rho_i T^{r\odot\varepsilon_i} \prod_m {\Sigma_m}^{-\varepsilon_{i,m}}\right)^2 = \left(\rho'_i T^{r\odot\varepsilon'_i} \prod_m {\Sigma_m}^{-\varepsilon'_{i,m}}\right)^2 \bmod N$$

telles que

$$\eta(\rho_i) \oplus (\varepsilon_i \odot r) \neq \eta(\rho'_i) \oplus (\varepsilon'_i \odot r).$$

On va donc étudier les variables aléatoires $\chi_i(\nu, r, \tau, f) = \eta(\rho_i) \oplus (\varepsilon_i \odot r)$.

Pour tout couple (i,j), on définit $c_{i,j}(\nu, r, \tau, f_j)$ par la valeur $c \in \{0,1\}$ maximisant

$$\Pr_f[((\nu, r, \tau, f) \in \mathcal{S}) \wedge (Ind_i = j) \wedge (\chi_i(\nu, r, \tau, f) = c) \,|\, f \text{ prolonge } f_j]$$

On peut alors définir les sous-ensembles $\mathcal{G}$ et $\mathcal{B}$ de $\mathcal{S}$ comme précédemment.

Pour tout $m \in \{1, \dots, k\}$, on définit la transformation Φ_m qui à tout quadruplet (ν, r, τ, f) associe le quadruplet (ν, r', τ', f) où

$$\begin{array}{rcl} r' & = & r_1 \dots r_{m-1}(\bar{r}_m) r_{m+1} \dots r_k = r \oplus 1_m, \\ \tau' & = & (t'_1, \dots, t'_\ell) \text{ avec } t'_i = t_i T^{e_{i,m}} \bmod N. \end{array}$$

Nous avons le même genre de résultat que précédemment au sujet de ces transformations Φ_m :

Lemme 97 .
Pour tout m, les exécutions correspondant à (ν, r, τ, f) et $\Phi_m(\nu, r, \tau, f)$ sont identiques pour l'attaquant.

Preuve. Soit $\mu \in \{1, \dots, k\}$ l'index d'une telle transformation, puis notons avec des «'» l'exécution correspondant à $\Phi_\mu(\nu, r, \tau, f)$. Nous n'avons qu'à vérifier que, pour tout i,

$y'_i = y_i$:

$$\begin{aligned} y'_i &= t'_i \prod_m (S_m^{(r')})^{e_{i,m}} = t'_i \prod_m \Sigma_m^{e_{i,m}} T^{r' \odot e_i} \bmod N \\ &= t_i \prod_m \Sigma_m^{e_{i,m}} T^{(r' \odot e_i) \oplus e_{i,\mu}} = t_i \prod_m \Sigma_m^{e_{i,m}} T^{(r' \odot e_i) \oplus (1_\mu \odot e_i)} \bmod N \\ &= t_i \prod_m \Sigma_m^{e_{i,m}} T^{((r' \oplus 1_\mu) \odot e_i)} = t_i \prod_m \Sigma_m^{e_{i,m}} T^{(r \odot e_i)} \\ &= t'_i \prod_m (S_m^{(r)})^{e_{i,m}} = y_i \bmod N. \end{aligned}$$

□

Lemme 98 .
Pour un triplet (ν, r, τ) fixé,

$$\Pr_f \left[((\nu, r, \tau, f) \in \mathcal{G}) \wedge \Big((\forall m)\ \Phi_m(\nu, r, \tau, f) \in \mathcal{G} \Big) \right] \leq Q^{\ell+1}/2^k.$$

Preuve. Comme dans la preuve du lemme 91, on suppose le contraire. Alors il existe un $\ell+1$-uplet $(u_1, \ldots, u_{\ell+1})$, un ℓ-uplet $(\tilde{e}_1, \ldots, \tilde{e}_\ell)$ ainsi que deux oracles aléatoires f et f' qui fournissent des réponses distinctes uniquement à certaines questions $\mathcal{Q}_{u_i} : \varepsilon_i \neq \varepsilon_i$. Notons i le plus petit de ces indices et $j = u_i$ l'indice de la question associée. Tous ces éléments sont tels que

$$\begin{array}{ll} (\nu, r, \tau, f) \in \mathcal{G} & \text{et } \Phi_m(\nu, r, \tau, f) \in \mathcal{G}, \text{ pour tout } m, \\ (\nu, r, \tau, f') \in \mathcal{G} & \text{et } \Phi_m(\nu, r, \tau, f') \in \mathcal{G}, \text{ pour tout } m. \end{array}$$

Pour tout m,

$$\begin{aligned} c_{i,j}(\nu, r, \tau, f_j) &= \eta(\rho_i(\nu, r, \tau, f)) \oplus (r \odot \varepsilon_i) = \eta(\rho_i(\Phi_m(\nu, r, \tau, f))) \oplus (r \odot \varepsilon_i) \\ &= c_{i,j}(\nu, r', \tau', f_j) \oplus \varepsilon_{i,m} \\ c_{i,j}(\nu, r, \tau, f'_j) &= \eta(\rho_i(\nu, r, \tau, f')) \oplus (r \odot \varepsilon'_i) = \eta(\rho_i(\Phi_m(\nu, r, \tau, f'))) \oplus (r \odot \varepsilon'_i) \\ &= c_{i,j}(\nu, r', \tau', f'_j) \oplus \varepsilon'_{i,m}. \end{aligned}$$

L'égalité entre les oracles partiels $f_j = f'_j$ entraîne $c_{i,j}(\nu, r, \tau, f_j) = c_{i,j}(\nu, r, \tau, f'_j)$. De plus, les challenges envoyés au signeur sont tous égaux à $(\tilde{e}_1, \ldots, \tilde{e}_\ell)$ donc les τ' sont égaux dans les deux cas. Ainsi, $c_{i,j}(\nu, r', \tau', f_j) = c_{i,j}(\nu, r', \tau', f'_j)$. Par conséquent, $\varepsilon'_{i,m} = \varepsilon_{i,m}$, et ce, pour tout m. Donc $\varepsilon'_i = \varepsilon_i$, ce qui contredit l'hypothèse. □

On en déduit :

$$\begin{aligned} \Pr[\mathcal{G}] &= \Pr[((\nu, r, \tau, f) \in \mathcal{G}) \wedge ((\forall m)\ \Phi_m(\nu, r, \tau, f) \in \mathcal{G})] \\ &+ \Pr[((\nu, r, \tau, f) \in \mathcal{G}) \wedge ((\exists m)\ \Phi(\nu, r, \tau, f) \in \mathcal{B})] \\ &\leq \frac{Q^{\ell+1}}{2^k} + \Pr[(\exists m)\ \Phi_m(\nu, r, \tau, f) \in \mathcal{B}] \\ &\leq \frac{Q^{\ell+1}}{2^k} + k \Pr[\mathcal{B}]. \end{aligned}$$

Alors, $\Pr[\mathcal{B}] \geq \frac{1}{k+1}\left(\Pr[\mathcal{S}] - \frac{Q^{\ell+1}}{2^k}\right) \geq \frac{1}{k+1}\left(\frac{3\varepsilon}{4} - \frac{Q^{\ell+1}}{2^k}\right)$. Pour n assez grand, nous pouvons espérer $Q^{\ell+1} \leq \varepsilon 2^k/4$. Dans ce cas, $\Pr[\mathcal{B}] \geq \frac{\varepsilon}{2(k+1)}$.

Nous savons donc que

$$\begin{aligned}
&\sum_{i=1}^{\ell+1} \Pr_{\nu,r,\tau,f}[(\chi_i(\nu,r,\tau,f) \neq c_{i,Ind_i}(\nu,r,\tau,f_{Ind_i})) \wedge \mathcal{S}] \\
&\geq \Pr_{\nu,r,\tau,f}[((\exists i)\ \chi_i(\nu,r,\tau,f) \neq c_{i,Ind_i}(\nu,r,\tau,f_{Ind_i})) \wedge \mathcal{S}] \\
&= \Pr[\mathcal{B}] \ \geq\ \frac{\varepsilon}{2(k+1)}.
\end{aligned}$$

Alors il existe $i \in \{1,\ldots,\ell+1\}$ tel que

$$\Pr[(\chi_i(\nu,r,\tau,f) \neq c_{i,Ind_i}(\nu,r,\tau,f_{Ind_i})) \wedge \mathcal{S}] \geq \frac{\varepsilon}{2(k+1)(\ell+1)}.$$

Si on choisit ce i au hasard, avec probabilité supérieure à $1/(\ell+1)$, on en choisit un «bon». Une machine semblable à celle décrite pour le schéma Okamoto–Schnorr a une probabilité de succès supérieure à $1/3$, lors de la première étape qui consiste a obtenir un succès en moins de $1/\varepsilon$ itérations. Dans ce cas, avec probabilité supérieure à $1/2(k+1)(\ell+1)$, $\chi_i \neq c_{i,Ind_i}$. Définissons l'ensemble

$$E = \left\{ (j,\nu,r,\tau,f_j) \,\middle|\, \Pr_f[\mathcal{S} \wedge (\chi_i \neq c_{i,j}) \wedge (Ind_i = j) \,|\, f \text{ prolonge } f_j] \geq \frac{\varepsilon}{4Q(k+1)(\ell+1)} \right\}.$$

Posons $j = Ind_i$. Avec probabilité supérieure à $1/2$, le succès obtenu vérifie de plus $(j,\nu,r,\tau,f_j) \in E$. Lors de la deuxième étape,

$$\Pr_{f'}\left[\mathcal{S} \wedge Ind_i = j \wedge \Big(\chi_i(\nu,r,\tau,f') \neq d\Big) \,\middle|\, f' \text{ prolonge } f_j\right] \geq \varepsilon/8(k+1)(\ell+1)Q.$$

Ainsi, après au plus $8(k+1)(\ell+1)Q/\varepsilon$ itérations, nous obtenons un deuxième succès exploitable avec probabilité supérieure à $1/3$. □

Théorème 99 (Signature en blanc Fiat-Shamir).
Supposons que la machine de Turing $\mathcal{C}$ soit capable d'effectuer une falsification supplémentaire selon une attaque parallèle en temps T, avec probabilité $\varepsilon \geq 1/P$, après Q appels à l'oracle aléatoire et ℓ appels à l'autorité. Si de plus, $\varepsilon \geq 4Q^{\ell+1}/2^k$, alors il existe une machine $\mathcal{M}'$ capable de fournir deux écritures distinctes d'un même élément, en temps $T' \leq 33Qk\ell T/\varepsilon$ et avec probabilité $\varepsilon' \geq 1/36k\ell^2$.

6.4.3 Autre «design» de signatures en blanc

À Eurocrypt '96, Shoup [109] a présenté un nouveau «design» de preuves de sécurité face aux attaques actives. Il l'a appliqué au schéma de Ong-Schnorr [78] et en fait une des seules alternatives, en 3 passes, aux schémas à «témoin indistinguable» d'Okamoto [77]. En fait, c'est un protocole à mi-chemin entre les protocoles à «témoin indistinguable» et à «divulgation nulle».

Un schéma de signature en blanc peut être dérivé de façon évidente de ce schéma, il est présenté figure 6.11. Au sujet de ce schéma, nous pouvons affirmer le résultat suivant :

Théorème 100 .
Dans le modèle de l'oracle aléatoire, une «falsification supplémentaire» du schéma de signature en blanc Ong-Schnorr, selon une «attaque séquentielle polylogarithmique» est équivalente à la factorisation des modules de Blum.

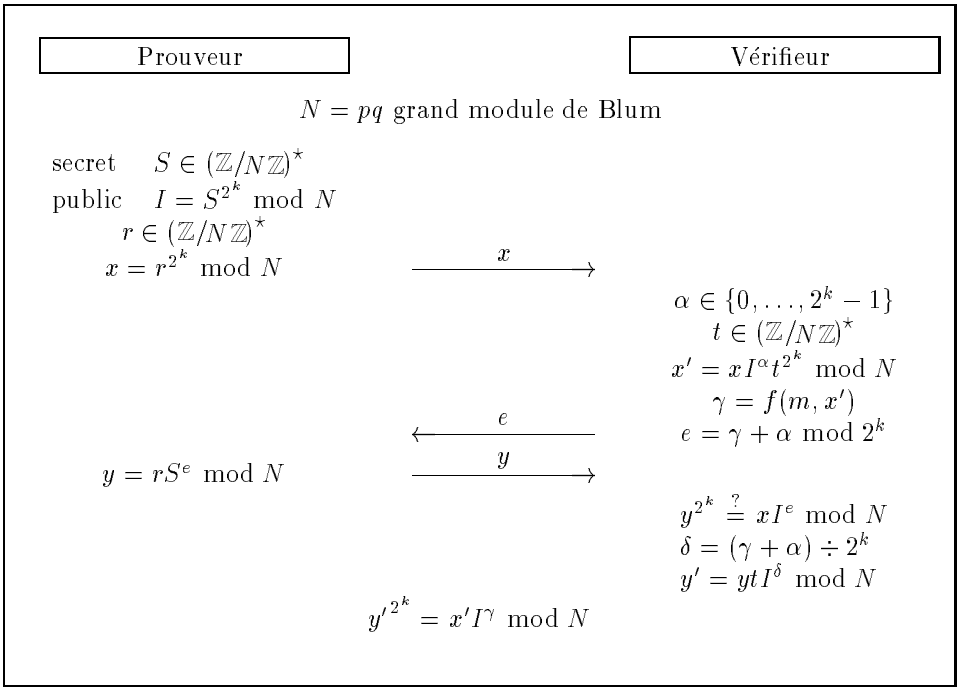

Fig. 6.11 – *Signature en blanc Ong-Schnorr*

Preuve. La démonstration de ce résultat est semblable à la démonstration du *lemme de bifurcation* 87. Cependant, la simulation de ce protocole n'a pas un succès garanti, elle ne permet alors pas au schéma de résister aux attaques parallèles.

Soit un attaquant qui, après ℓ interactions séquentielles avec le signeur, parvient à fabriquer $\ell + 1$ signatures valides avec probabilité non négligeable $\varepsilon > 1/P$. Posons $\lambda = \lceil \log(1/\varepsilon) \rceil + 1$.

On répond aux ℓ interactions à l'aide du simulateur présenté figure 3.7, possédant une «pseudo-clé secrète», en effectuant un «reset» de l'attaquant lors des échecs. Puis l'attaquant parvient à fabriquer $\ell + 1$ signatures avec une probabilité $\varepsilon > 1/2^{\lambda-1}$.

Comme dans la preuve de sécurité du schéma d'identification, on fait «bifurquer» l'attaque de sorte à obtenir deux signatures (m, x, e, y) et (m, x, e', y') avec $e' \neq e \bmod 2^{\lambda+1}$. Nous obtenons alors, de la même manière, avec probabilité un demi, un facteur de N.

□

6.4.4 Conclusion

Avant de conclure, nous dirons quelques mots sur les «preuves transférables» (*transferable proofs* et *divertible proofs*) [75, 57, 15, 28]. À la manière de la «fraude de la mafia» [30], elles peuvent être transformées en schémas de signature en blanc. Cependant, les résultats de sécurité prouvés pour ces schémas peuvent s'exprimer de la façon suivante [27, 28] : si le protocole d'identification de base est à «témoin caché», alors aucune $(1, 2)$-falsification ne peut avoir une probabilité de succès écrasante en temps polynomial.

La propriété de «témoin indistinguable» semble donc être la notion appropriée pour fabriquer des schémas de signature en blanc sûrs, notamment contre les attaques parallèles. Un affaiblissement de cette propriété en adaptant un protocole à 3 passes simplement sûr contre les attaques actives peut ne plus garantir la même sécurité.

Chapitre 7

Monnaie électronique

7.1 Définition

Avec l'importance croissante d'Internet et du commerce mondial, la monnaie électronique est devenue une branche active de la recherche en cryptographie. Les notions cryptographiques sur lesquelles repose la monnaie électronique ont été introduites par David Chaum [19, 20, 21]. Dès 1982, son objectif était de produire une version électronique de l'argent liquide, et donc, avec les mêmes propriétés :

- tout d'abord, il souhaitait l'anonymat. En effet, dans la vie réelle, rien ne ressemble plus à une pièce de un franc qu'une autre pièce de un franc. Ainsi, une pièce remise par la banque à Alice, dépensée chez le boulanger $\mathcal{B}$ et redéposée à la banque ne pourra être reconnue par la banque. Dans le cas contraire, la banque apprendrait qu'Alice achète son pain chez $\mathcal{B}$. Pour cela, on dit qu'une pièce ne peut être «tracée». Une autre forme d'anonymat est de ne pas pouvoir «relier» deux pièces comme appartenant à la même personne.

- une autre nécessité est le contrôle de la quantité d'argent en circulation par la banque.

Il affirma que le seul moyen de remplir ces propriétés était l'utilisation de «pièces» associées à la notion de «signatures en blanc».

Quand un utilisateur retire de l'argent de la banque, la banque signe «en blanc» les pièces créées par l'utilisateur lui-même. L'utilisateur peut alors les dépenser chez un commerçant qui finalement dépose les «pièces» à la banque (voir figure 7.1).

Dans son premier schéma, David Chaum utilisait les «signatures en blanc» pour la production de pièces. L'utilisateur fait signer en blanc par la Banque une pièce. Il se trouve alors en possession d'une pièce valide que la Banque elle-même est incapable de reconnaître. Lors de la dépense de cette pièce, le commerçant la renvoie immédiatement à la Banque. La Banque vérifie alors que cette pièce est utilisée pour la première fois et informe le commerçant de la validité de la pièce. Le parcours de la pièce (voir figure 7.1) ainsi décrit restera le même tout au cours de ce chapitre. Cependant, pour ce premier schéma, nous nous plaçons dans un contexte «on-line», c'est-à-dire que le commerçant est en communication permanente avec la Banque pour avoir l'information sur la validité de la pièce lors de sa dépense. Pour ce premier schéma, David Chaum définit le premier

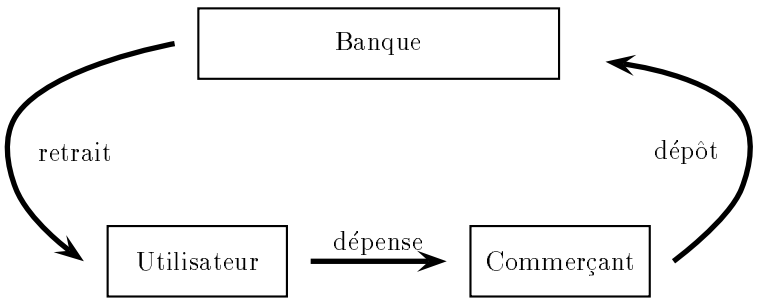

Fig. 7.1 – *Parcours d'une pièce*

schéma de signature en blanc, basé sur le schéma de signature RSA, décrit au début du chapitre précédent.

Il serait tout de même préférable de pouvoir passer dans un contexte «off-line» pour limiter les communications et simplifier la mise en œuvre. Mais il est impossible d'empêcher un utilisateur de dupliquer des pièces qu'il possède et de les dépenser plusieurs fois. Cette fraude est appelée «double dépense» ou plus généralement «dépense multiple». À défaut d'empêcher une «double dépense», Chaum, Fiat et Naor [25] ont eu l'idée de déterminer l'identité du fraudeur. Plus précisément, ils introduisirent l'identité de l'utilisateur dans la pièce de telle sorte qu'en cas d'utilisation normale la pièce reste anonyme, mais en cas de «double dépense» l'identité soit révélée.

Encore une fois, la signature en blanc, ici RSA, est cruciale pour garantir l'anonymat. Pour garantir la présence de l'identité dans la pièce, créée par l'utilisateur lui-même, ils eurent recours à la technique du «cut-and-choose» :

- La Banque possède un grand module RSA N public, une clé publique $e = 3$ et la clé secrète s associée. Les fonctions f et g seront à sens-unique et publiques.

- Une pièce de monnaie est le produit π de k entiers t_i, qui sont des signatures en blanc de $f(x_i, y_i)$, de la forme $f(x_i, y_i)^s \bmod N$, où k est le facteur de sécurité. De plus, pour tout indice i, $x_i = g(a_i, c_i)$ et $y_i = g(b_i, c_i \oplus I)$, où a_i, b_i et c_i sont des valeurs aléatoires, et I l'identité de l'utilisateur.

$$\pi = \prod_{i=1}^{k} t_i = \prod_{i=1}^{k} f(x_i, y_i)^s \bmod N.$$

- Pour dépenser une telle pièce, Alice reçoit un challenge $d \in \{0,1\}^k$ du commerçant Bob. Pour chaque indice i, Alice répond

  $d_i = 0$: Alice envoie a_i, c_i et y_i. Bob peut calculer $x_i = g(a_i, c_i)$ et $\tau_i = f(x_i, y_i)$.

  $d_i = 1$: Alice envoie b_i, $c_i \oplus I$ et x_i. Bob peut calculer $y_i = g(b_i, c_i \oplus I)$ et $\tau_i = f(x_i, y_i)$.

  Enfin, Bob vérifie si $\pi^3 = \prod \tau_i \bmod N$. Seule la Banque était capable de fabriquer un tel produit.

- En cas de «double dépense», avec forte probabilité, deux challenges différents sont posés. Cela signifie qu'il existe i tel que $d_i \neq d'_i$. Alors Alice doit révéler c_i et $c_i \oplus I$. L'anonymat disparaît.

- Pour garantir la détection de l'auteur d'une «double dépense», une pièce doit respecter le format précédemment décrit. En particulier, l'identité doit être correcte. Un fraudeur n'aura aucune raison d'être honnête et de respecter ce format. La Banque doit donc contrôler le format des pièces sans pour autant en violer le secret. Pour cela, la Banque fournit $2k$ signatures en blanc pour permettre à Alice de fabriquer deux fois plus de t_i que nécessaire. Parmi les $2k$ nombres t_i produits, la Banque en choisit k pour vérifier la validité de leur structure. Ayant perdu leur caractère anonyme, ces k nombres seront jetés. Les k restants permettront la construction de la pièce. La probabilité pour un fraudeur de se trouver en possession d'une pièce contenant une identité erronée est de l'ordre de 2^{-2k}.

Les inconvénients de cette technique sont l'accroissement des communications lors des retraits ainsi que la taille volumineuse des pièces. De nombreuses améliorations suivirent [24, 76] et, en 1993, apparurent les premiers schémas sans «cut-and-choose» [9, 8, 40, 39]. Mais plus récemment, la notion d'anonymat inconditionnel est critiquée en raison du blanchiment d'argent et de divers autres crimes [67] : la notion d'anonymat conditionnel ou «révocable», qui peut être levée par une autorité, est un nouveau moteur. De nombreux articles [58, 66] sont déjà parus à ce sujet. Une autre possibilité est l'utilisation de signatures en blanc à trappe [17, 18, 42, 16] qui permettraient à une autorité de lever l'anonymat si nécessaire. Mais nous ne nous intéresserons dans cette thèse qu'à l'anonymat parfait.

7.2 Schéma de Brands

7.2.1 Certificat de clé secrète

Récemment, Brands [14, 12] a défini la notion de «secret-key certificate» (ou *certificat de clé secrète*). Ce certificat est un triplet (h, e, ρ) qui constitue une preuve de connaissance d'une clé secrète associée à h. De plus, cette preuve est certifiée par une autorité. Il a ensuite utilisé ces certificats en monnaie électronique. Son premier exemple (voir figure 7.2) provient du schéma de signature de Schnorr. Il vérifie, par définition, les deux propriétés suivantes, l'indistinguabilité et l'infalsifiabilité.

Indistinguabilité : Il doit être possible de simuler des triplets (h, e, ρ) avec une distribution indistinguable de celle suivie par les triplets réellement fabriqués suite à une communication avec l'autorité.

Pour cela, on choisit deux entiers $t_1, t \in \mathbb{Z}/q\mathbb{Z}$, on calcule $h = h_0^{-1} g^{t_1} \bmod p$ ainsi que $a = g^t \bmod p$. Alors $e = H(h, a)$ et $\rho = t - et_1 \bmod q$ fournissent un certificat correct de h.

Infalsifiabilité : En revanche, une telle simulation fournit un triplet inexploitable, dans le sens où il est impossible de prouver la connaissance d'une clé secrète associée à h,

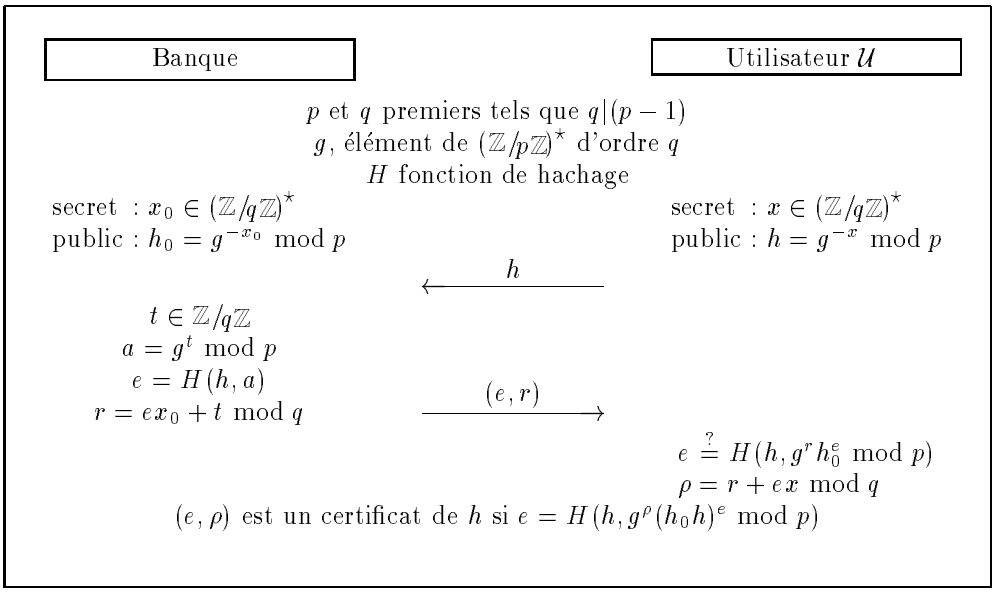

Fig. 7.2 – *Certificat de clé secrète – Schnorr*

à moins de connaître celle de h_0. Il doit en être de même pour tout autre tentative de fabrication de certificats sans l'aide de l'autorité.

Supposons que Charlie soit capable de fabriquer de tels triplets (falsification existentielle), ou un nouveau triplet après avoir interrogé l'autorité (attaque à messages choisis adaptative), avec probabilité non négligeable. En appliquant le *lemme de bifurcation* et la technique de l'«oracle replay» sur une coalition entre Charlie et un simulateur de l'autorité, nous obtenons deux certificats valides (h, a, e, ρ) et (h, a, e', ρ') où $e \neq e'$ et $h = g^{-x} \bmod p$, avec probabilité non négligeable. Ainsi,

$$a = g^\rho (hh_0)^e = g^{\rho'} (hh_0)^{e'} \bmod p.$$

Alors, en posant $y = (\rho - \rho')/(e' - e) \bmod q$, il vient $h_0 = g^{y+x} \bmod p$.

7.2.2 Application à la monnaie électronique

Brands a eu l'idée de fabriquer une version «en blanc» de ces certificats pour en faire des pièces de monnaie, en s'arrangeant pour qu'une «double dépense» révèle la clé secrète contenant l'identité.

Retrait (voir figure 7.3). Comme dans tout schéma à base de logarithme discret, on a deux grands nombres premiers p et q tels que $q|p-1$, ainsi qu'un élément g de $(\mathbb{Z}/p\mathbb{Z})^\star$ d'ordre q. La Banque choisit deux éléments, x_0 et λ de $(\mathbb{Z}/q\mathbb{Z})^\star$, qu'elle garde secrets, et publie $g_1 = g^\lambda \bmod p$ ainsi que $h_0 = g^{-x_0} \bmod p$. Tout utilisateur choisit une clé secrète $x_\mathcal{U}$ liée à son identité $\mathcal{U}$ et calcule sa clé publique $h_\mathcal{U} = g_1^{-x_\mathcal{U}} \bmod p$. La Banque a également connaissance de la clé secrète $x_\mathcal{U}$ de tout utilisateur.

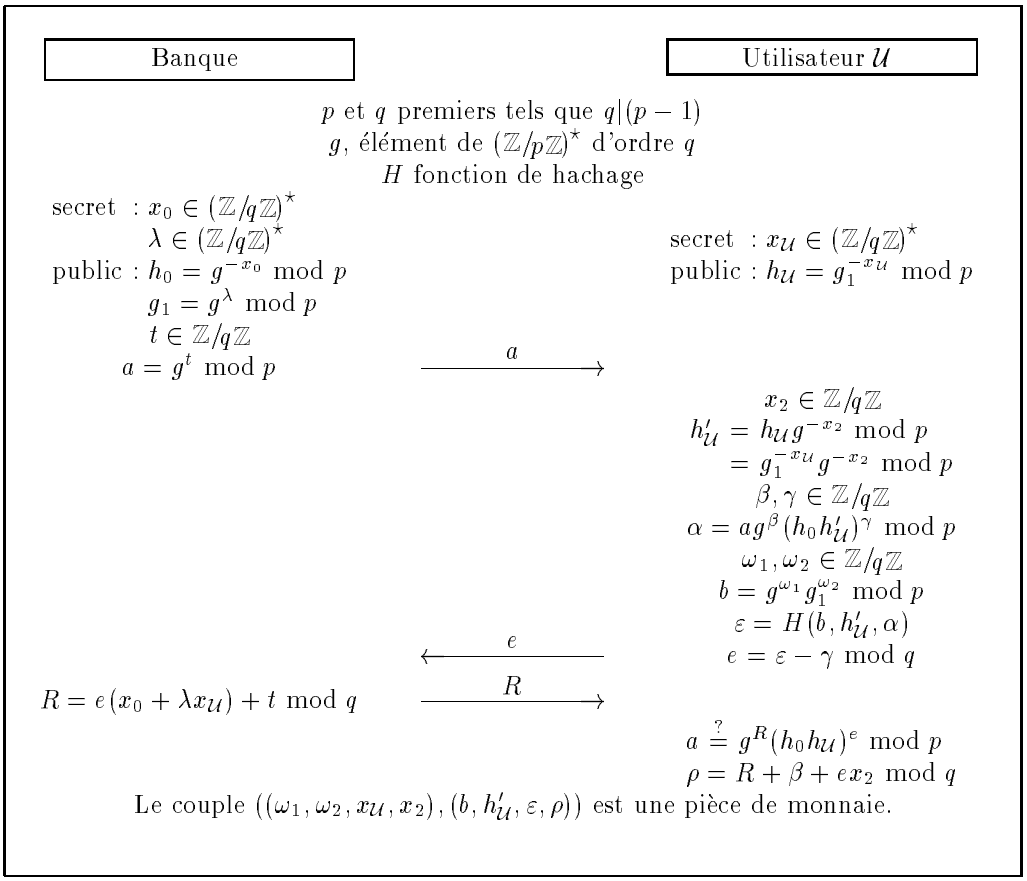

Fig. 7.3 – *Retrait de Brands*

Une pièce de monnaie sera un couple $((\omega_1, \omega_2, x_{\mathcal{U}}, x_2), (b, h'_{\mathcal{U}}, \varepsilon, \rho))$ où la première partie est gardée secrète par l'utilisateur, tel que (ε, ρ) soit un certificat de $(b, h'_{\mathcal{U}})$, c'est-à-dire une signature en blanc de $(b, h'_{\mathcal{U}})$ vérifiant $\varepsilon = H(b, h'_{\mathcal{U}}, g^\rho (h_0 h'_{\mathcal{U}})^\varepsilon \bmod p)$. De plus l'utilisateur est capable de prouver sa connaissance de la clé secrète $(x_{\mathcal{U}}, x_2)$ associée à $h'_{\mathcal{U}}$.

Dépense (voir figure 7.4). La dépense consiste à fournir la deuxième partie de la pièce $(b, h'_{\mathcal{U}}, \varepsilon, \rho)$ afin que le commerçant vérifie la validité du certificat, puis $\mathcal{U}$ prouve sa connaissance d'une clé secrète associée à $h'_{\mathcal{U}}$ à l'aide du protocole d'Okamoto–Schnorr, avec la mise en gage b.

Dépôt Le commerçant n'a plus qu'à aller présenter la pièce $(b, h'_{\mathcal{U}}, \varepsilon, \rho)$ avec la preuve de connaissance de la clé secrète associée, (m, φ, C, R, S), où m est le montant, φ un aléa, et C son identité, à la Banque.

On remarque que cette pièce est en fait une signature en blanc de la clé secrète, et donc est totalement indépendante de l'identité de l'utilisateur pour une utilisation standard. L'anonymat est ainsi inconditionnellement conservé tout au long du parcours de la pièce. En revanche, en cas de «double dépense», $x_{\mathcal{U}}$ et x_2 sont révélés, fournissant l'identité du fraudeur.

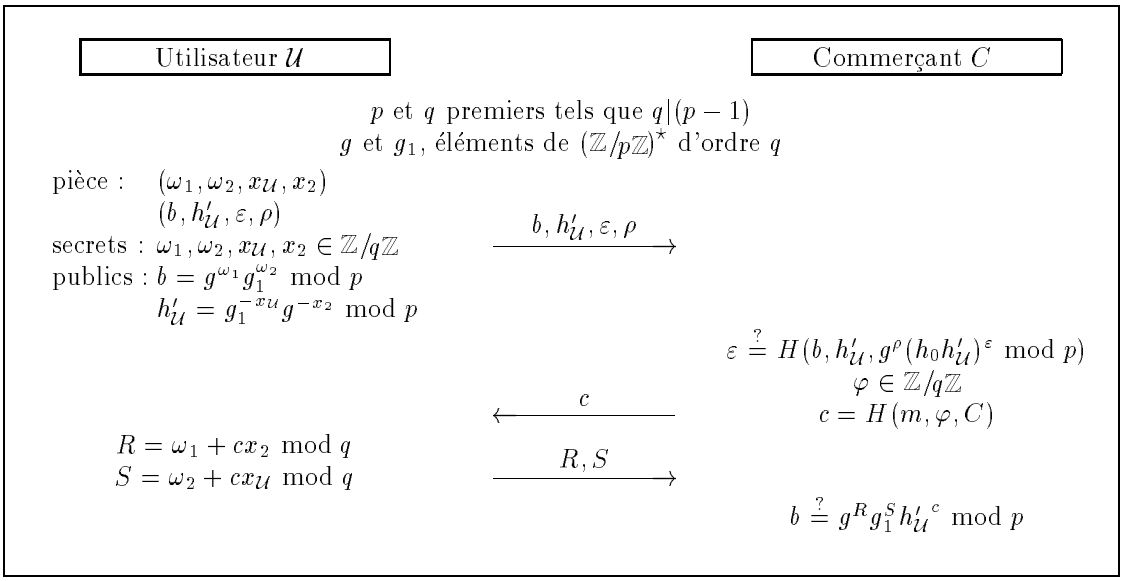

Fig. 7.4 – *Dépense de Brands*

7.2.3 Sécurité

Pour valider son schéma, Brands a avancé deux hypothèses de sécurité plus ou moins explicites au sujet du contrôle de la quantité d'argent en circulation par la Banque et de la garantie de retrouver un fraudeur. Nous allons montrer qu'elles ne sont pas satisfaites par son schéma, ou tout du moins prouvées, et nous nous efforcerons de proposer une réparation les vérifiant.

7.2.3.1 Contrôle de la quantité d'argent

La première hypothèse est qu'après ℓ retraits séquentiels ou parallèles, l'utilisateur ne peut posséder plus de ℓ pièces. C'est d'ailleurs suite à cette supposition que nous avons défini la notion de falsification supplémentaire pour les signatures en blanc. Malheureusement, aucune preuve ne permet de valider une telle hypothèse pour ce schéma. En effet, les seules preuves de sécurité existantes pour les signatures en blanc sont celles présentées au chapitre 6 qui ne s'appliquent qu'aux schémas à base de protocoles «à témoin indistinguable», ce qui n'est pas le cas du schéma de Schnorr.

7.2.3.2 Double dépense

Sa deuxième hypothèse implicite est que chaque pièce contient obligatoirement l'identité de l'utilisateur. En effet, pour contrer une «double dépense», il faut être sûr de la présence de l'identité réelle de l'utilisateur dans la pièce. Or, par la suite, Brands a lui-même mis en évidence une faille face à un retrait parallèle de plusieurs personnes [13]. Mais en fait, un unique utilisateur peut également fabriquer, après deux retraits en parallèles, une pièce contenant une identité totalement aléatoire (voir figure 7.5).

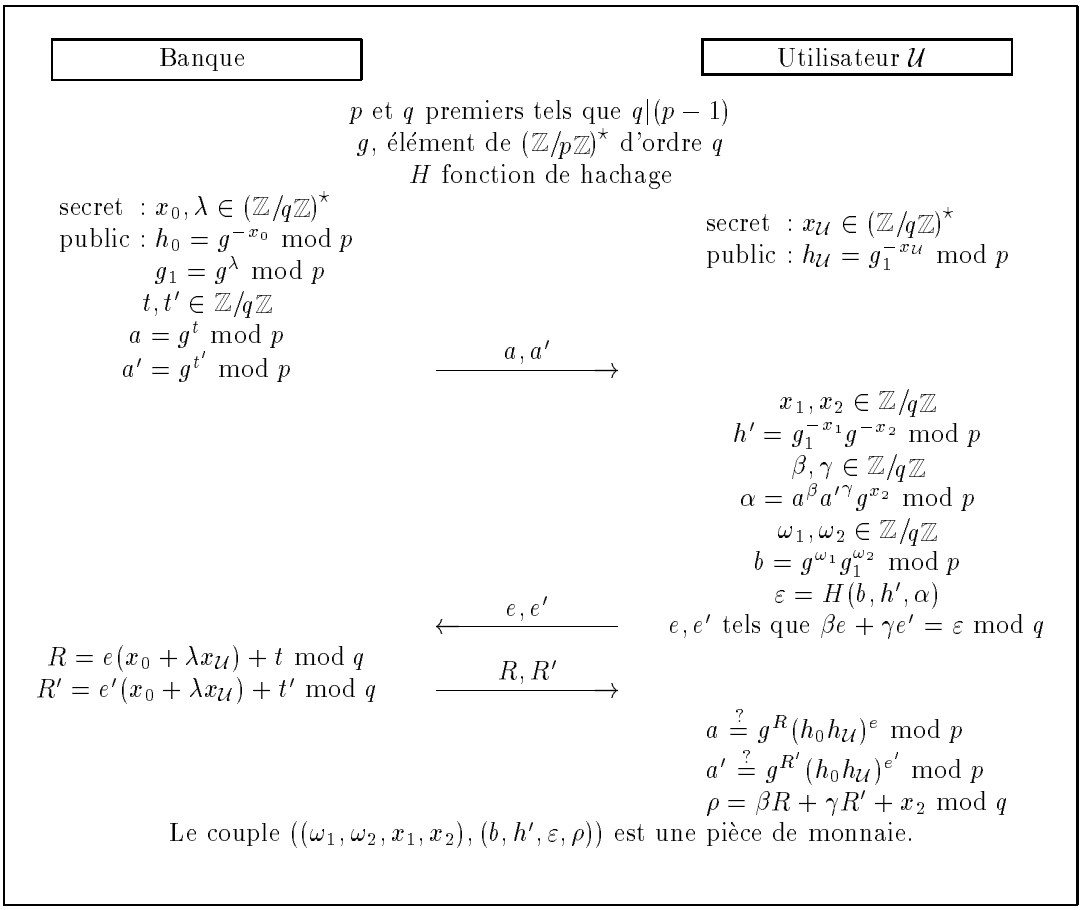

Fig. 7.5 – *Fraude parallèle du schéma de Brands*

7.2.4 Réparation

Brands a tenté une première fois de réparer son protocole, mais en vain [10, 13]. Puis, suite à une remarque de Schoenmakers [101], il a proposé un nouveau schéma plus «sûr» du même genre [11]. Cependant, dans leurs articles [11, 101], Brands et Schoenmakers ont simplement prouvé que précisément l'attaque parallèle précédemment remarquée ne fonctionnait plus. Rien n'assure qu'une autre attaque parallèle, voire séquentielle, ne permettrait pas le même genre d'abus. Quant à la falsification supplémentaire, aucune preuve n'est proposée.

7.3 Nouveau schéma avec sécurité prouvée

Nous allons donc présenter un nouveau schéma possédant une sécurité prouvée face aux deux fraudes précédemment énoncées :

– en utilisant un schéma de signature en blanc prouvé sûr, nous allons garantir l'impossibilité d'une falsification supplémentaire. La Banque aura donc le contrôle total de l'argent en circulation.

– aucune fraude séquentielle ou parallèle d'un utilisateur isolé ne permettra de fabriquer une pièce sans son identité. La sécurité contre une fraude séquentielle ou parallèle d'un groupe d'utilisateurs restera à l'état de conjecture.

Des garanties pour l'utilisateur seront également apportées.

7.3.1 Protection contre la «double dépense»

Tout d'abord, nous allons tenter de réparer les schémas de Brands et Schoenmakers [12, 13, 11, 101] avec preuve de sécurité à l'appui contre toute attaque d'un utilisateur isolé. La première version (simplifiée) du schéma de retrait (voir figure 7.6) est inspirée du schéma de Schoenmakers [101]. Ce schéma, comme celui de Schoenmakers, ressemble au schéma original de Brands mais avec un certificat de clé secrète différent. L'équation à satisfaire est cette fois-ci, $\alpha = g'^{\rho} y_0^{\varepsilon} \bmod p$, tout en connaissant une écriture de g' dans la base (g_1, g_2).

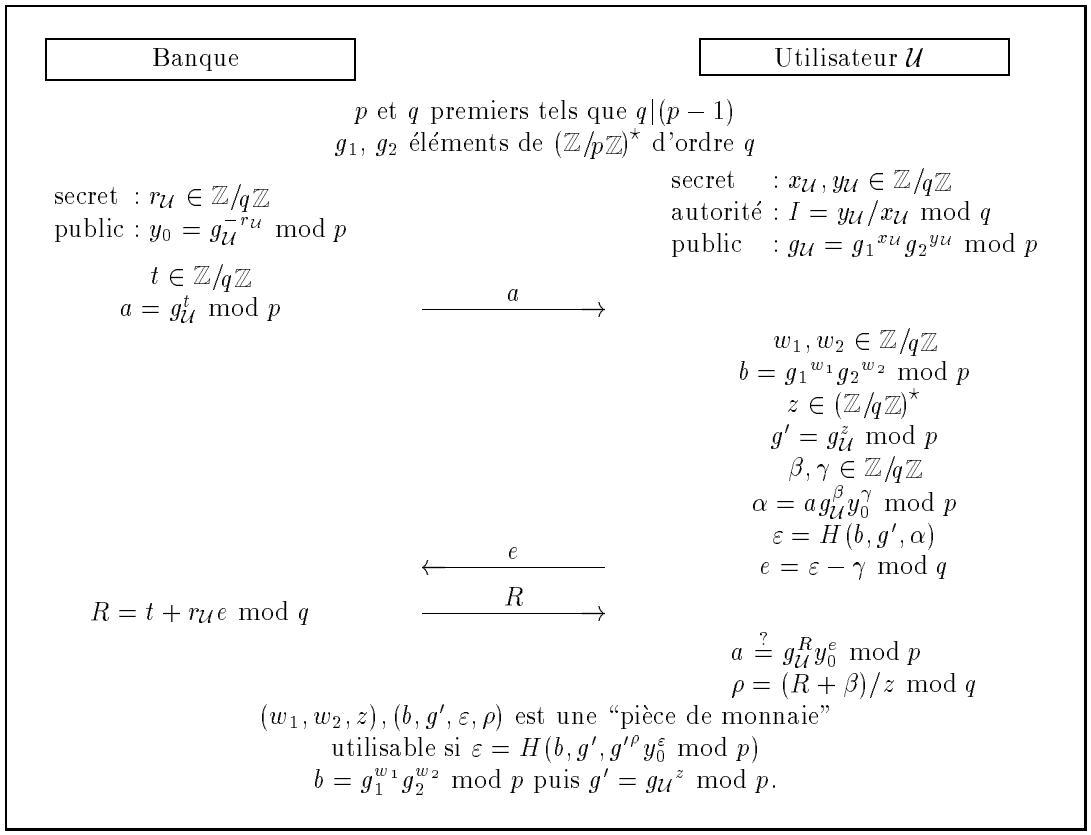

Fig. 7.6 – *Retrait simplifié*

La dépense d'une telle pièce est semblable à celle déjà présentée. C'est-à-dire que l'utilisateur $\mathcal{U}$ fournit le quadruplet $(b, g', \varepsilon, \rho)$ avec la preuve de connaissance d'un couple (r, s) associé à g' avec le protocole d'Okamoto-Schnorr et une mise en gage b. Ainsi, en cas de «double dépense», avec forte probabilité, deux «challenges» distincts seront demandés et feront révéler à $\mathcal{U}$ la clé secrète (r, s). Si la pièce a été fabriquée honnêtement, le rapport

s/r est égal à l'identifiant $I_{\mathcal{U}}$ associé à l'utilisateur. Il est donc important, pour contrer une «double dépense», que g' «contienne» l'identifiant $I_{\mathcal{U}}$.

Théorème 101 (Sécurité face à une personne).
Une usurpation d'identité, même selon une attaque parallèle, d'une seule personne est équivalente au logarithme discret.

Preuve. L'analyse est semblable à la preuve de sécurité des schémas de signature en blanc. Charlie $\mathcal{C}$ initie en parallèle le retrait de ℓ pièces et tente de fabriquer une pièce contenant une identité I' distincte de la sienne $I_{\mathcal{C}}$. Il pourra ainsi l'utiliser à volonté sans crainte d'être poursuivi. Charlie est référencé auprès d'une Banque avec une clé secrète $(x_{\mathcal{C}}, y_{\mathcal{C}})$ et une clé publique $g_{\mathcal{C}} = g_1^{x_{\mathcal{C}}} g_2^{y_{\mathcal{C}}} \bmod p$. Il est identifié par l'autorité sous $I_{\mathcal{C}} = y_{\mathcal{C}}/x_{\mathcal{C}} \bmod q$.

La Banque possède une clé publique globale y_0 et une clé secrète $r_{\mathcal{C}}$ liée à Charlie. À la demande de ℓ pièces, la Banque envoie $a_1, \ldots, a_\ell$. Charlie pose alors un certain nombre de questions, $\mathcal{Q}_1, \ldots, \mathcal{Q}_Q$, à l'oracle aléatoire tout en envoyant les «challenges» e_i à la Banque qui lui retourne R_i satisfaisant $a_i = {g_{\mathcal{C}}}^{R_i} y_0^{e_i} \bmod p$. Charlie parvient alors à fabriquer une pièce valide, c'est-à-dire deux couples (w_1, w_2) et (r', s') correspondants à b et g', puis un certificat (ε, ρ) satisfaisant :

$$b = g_1^{w_1} g_2^{w_2} \bmod p, \qquad g' = g_1^{r'} g_2^{s'} \bmod p,$$
$$\alpha = {g'}^{\rho} y^{\varepsilon} \bmod p, \qquad \varepsilon = H(b, g', \alpha).$$

Les deux premiers couples ne font pas partie de la fraction publique de la pièce, mais ils peuvent être obtenus lors d'une «double dépense». On peut donc les considérer liés à la pièce. En effet, si plusieurs couples distincts peuvent être associés à une même pièce, le logarithme discret de g_2 relativement à g_1 peut être déterminé, alors qu'il est *a priori* inconnu. On suppose de plus que l'identité contenue dans la pièces est distincte de celle de l'attaquant, cela signifie que $I_{\mathcal{C}} = y_{\mathcal{C}}/x_{\mathcal{C}} \neq s'/r' = I'$.

On regroupera sous la variable ν le ruban aléatoire ω de $\mathcal{C}$, ainsi que y_0 et $(a_1, \ldots, a_\ell)$. Une exécution de l'attaque sera représentée par le triplet $(\nu, r_{\mathcal{C}}, f)$, où f est l'oracle aléatoire. Charlie est un attaquant efficace, c'est-à-dire qu'il existe un polynôme P tel que

$$\Pr_{\nu, r_{\mathcal{C}}, f}\left[\mathcal{C} \text{ fournit } (w_1, w_2), (r', s'), b, g', (\varepsilon, \rho) \text{ satisfaisant } I_{\mathcal{C}} \neq \frac{s'}{r'}\right] \geq \varepsilon \geq 1/P.$$

En appliquant la technique du «lemme de bifurcation» et de l'«oracle replay», on obtient deux pièces avec une partie commune, (w_1, w_2), (r', s'), b et g', mais $(\varepsilon, \rho) \neq (\varepsilon', \rho')$. Les deux pièces ainsi obtenues contiennent donc l'identifiant $I' = s'/r' \neq I_{\mathcal{C}}$.

Si on définit les variables aléatoires $\chi = \rho r' - \varepsilon x_{\mathcal{C}} r_{\mathcal{C}} \bmod q$ et $\psi = \rho s' - \varepsilon y_{\mathcal{C}} r_{\mathcal{C}} \bmod q$, on obtient $\psi - I'\chi = (I' - I_{\mathcal{C}})\varepsilon r_{\mathcal{C}} x_{\mathcal{C}} \bmod q$, en éliminant ρ. Et alors, la validité des certificats se traduit par les égalités

$$\begin{aligned} \alpha &= {g'}^{\rho} y^{\varepsilon} = g_1^{\rho r' - \varepsilon x_{\mathcal{C}} r_{\mathcal{C}}} g_2^{\rho s' - \varepsilon y_{\mathcal{C}} r_{\mathcal{C}}} \bmod p = g_1^{\chi} g_2^{\psi} \bmod p \\ \alpha &= {g'}^{\rho'} y^{\varepsilon'} = g_1^{\rho' r' - \varepsilon' x_{\mathcal{C}} r_{\mathcal{C}}} g_2^{\rho' s' - \varepsilon' y_{\mathcal{C}} r_{\mathcal{C}}} \bmod p = g_1^{\chi'} g_2^{\psi'} \bmod p. \end{aligned}$$

Si l'identité I' est distincte de $I_{\mathcal{C}}$, alors $\psi - I'\chi \neq \psi' - I'\chi' \bmod q$. Ceci entraîne $\chi \neq \chi'$ ou $\psi \neq \psi'$, et fournit une égalité non triviale, $g_1^{\chi} g_2^{\psi} = g_1^{\chi'} g_2^{\psi'} \bmod p$. On en déduit alors le logarithme de g_2 relativement à g_1. □

Il est bien évident que cette preuve ne convient plus pour une attaque à plusieurs individus. En effet, pour fabriquer des clés valides $(x_{\mathcal{U}}, y_{\mathcal{U}}, r_{\mathcal{U}})$ et $(x_{\mathcal{V}}, y_{\mathcal{V}}, r_{\mathcal{V}})$ pour deux individus $\mathcal{U}$ et $\mathcal{V}$, l'autorité doit connaître le logarithme de g_2 relativement à g_1 :

$$y_0 = g_{\mathcal{U}}^{-r_{\mathcal{U}}} = g_1^{-r_{\mathcal{U}} x_{\mathcal{U}}} g_2^{-r_{\mathcal{U}} y_{\mathcal{U}}} = g_{\mathcal{V}}^{-r_{\mathcal{V}}} = g_1^{-r_{\mathcal{V}} x_{\mathcal{V}}} g_2^{-r_{\mathcal{V}} y_{\mathcal{V}}} \bmod p.$$

Pour ne rien révéler, il faut que les deux représentations soient identiques, c'est-à-dire $r_{\mathcal{U}} x_{\mathcal{U}} = r_{\mathcal{V}} x_{\mathcal{V}} \bmod q$ et $r_{\mathcal{U}} y_{\mathcal{U}} = r_{\mathcal{V}} y_{\mathcal{V}} \bmod q$. Et donc, nécessairement, $I_{\mathcal{U}} = I_{\mathcal{V}} \bmod q$, ce qui n'en fait plus un identifiant. Mais l'idée essentielle du schéma final est présentée.

7.3.2 Protection contre la création d'argent

Nous allons maintenant protéger notre système contre la création frauduleuse d'argent. Brands et Schoenmakers ont tous deux utilisé un certificat à base de Schnorr, ce qui ne permet pas de garantir qu'après ℓ interactions avec la Banque, un utilisateur ne peut posséder plus de ℓ pièces. Nous allons donc présenter un nouveau certificat de clé secrète à base du schéma d'Okamoto–Schnorr (voir figure 7.7).

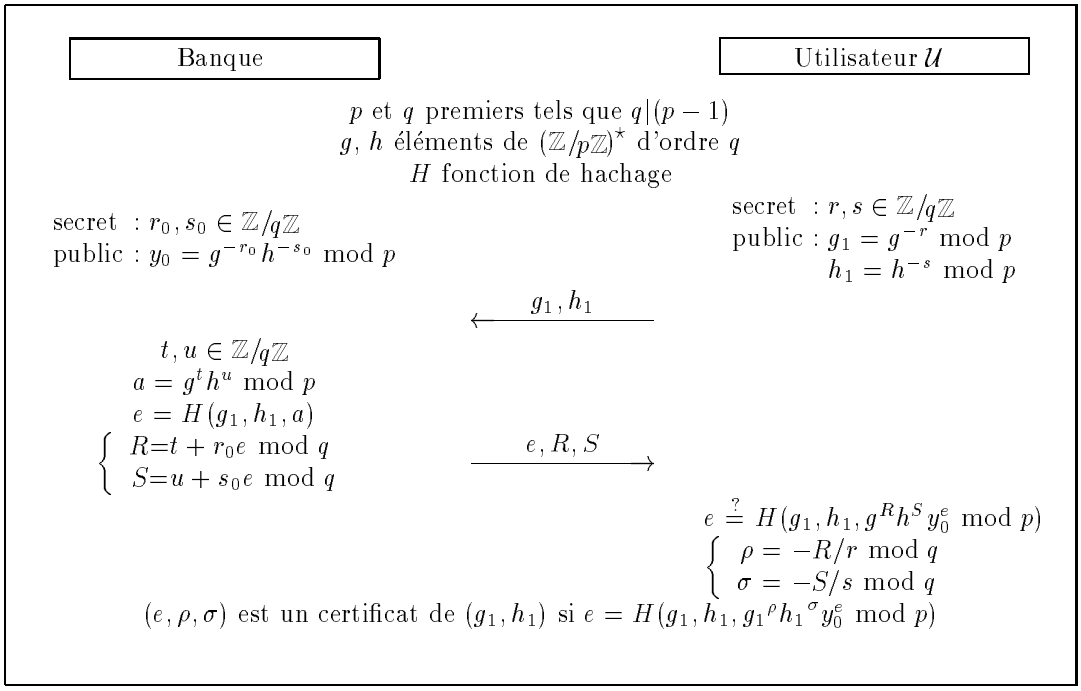

Fig. 7.7 – *Certificat de clé secrète – Okamoto–Schnorr*

Indistinguabilité Montrons qu'une simulation d'un tel certificat est aisée, avec une distribution indistinguable de celle produite suite à une véritable communication avec

l'autorité. On choisit des éléments t, u, t_1, u_1, t_2 et u_2 aléatoirement dans $\mathbb{Z}/q\mathbb{Z}$ tels que $\Delta = u_1t_2 - u_2t_1 \neq 0 \bmod q$ et on pose

$$\begin{cases} a &= g^t h^u \bmod p, \\ g_1 &= a^{t_1} y_0^{u_1} \bmod p, \\ h_1 &= a^{t_2} y_0^{u_2} \bmod p. \end{cases}$$

Ensuite, on calcule $e = H(g_1, h_1, a)$, et on résout le système

$$\begin{cases} 1 &= t_1\rho + t_2\sigma \bmod q \\ -e &= u_1\rho + u_2\sigma \bmod q. \end{cases}$$

Son déterminant est inversible alors,

$$\begin{cases} \rho &= -(u_2 + et_2)/\Delta \bmod q \\ \sigma &= (u_1 + et_1)/\Delta \bmod q. \end{cases}$$

Par suite, nous avons, ${g_1}^{\rho} {h_1}^{\sigma} = a^{t_1\rho + t_2\sigma} y_0^{u_1\rho + u_2\sigma} = a y_0^{-e} \bmod p$ avec $e = H(g_1, h_1, a)$. Mais dans cette simulation, l'utilisateur est incapable de prouver sa connaissance des décompositions de g_1 et h_1 relativement à g et h.

Infalsifiabilité Supposons que Charlie soit capable de fabriquer de tels certificats valides (falsification existentielle), ou un nouveau après avoir interrogé l'autorité (attaque à messages choisis adaptative), avec probabilité non négligeable. En appliquant le *lemme de bifurcation* et la technique de l'«oracle replay» sur une coalition entre Charlie et l'autorité possédant une clé secrète (r_0, s_0) quelconque, nous obtenons deux certificats valides $(g_1, h_1, a, e, \rho, \sigma)$ et $(g_1, h_1, a, e', \rho', \sigma')$ où $e \neq e'$ et $g_1 = g^{-r} \bmod p$ et $h_1 = h^{-s} \bmod p$, avec probabilité non négligeable. Ainsi,

$$a = {g_1}^{\rho} {h_1}^{\sigma} y_0^e = g^{-r\rho - r_0 e} h^{-s\sigma - s_0 e} = {g_1}^{\rho'} {h_1}^{\sigma'} y_0^{e'} = g^{-r\rho' - r_0 e'} h^{-s\sigma' - s_0 e'} \bmod p.$$

Alors, comme pour les schémas de signature en blanc, en étudiant les variables aléatoires

$$\begin{cases} \varphi &= r\rho + r_0 e \bmod q, \\ \psi &= s\sigma + s_0 e \bmod q, \end{cases}$$

on peut montrer que l'égalité $g^{\varphi' - \varphi} = h^{\psi - \psi'} \bmod p$ est non triviale avec bonne probabilité. Ceci fournit le logarithme de h relativement à g.

7.3.3 Retrait d'argent

Nous allons maintenant «mixer» ces deux techniques afin de fabriquer des pièces de monnaies sûres contre de nombreuses attaques. L'utilisation de ce certificat «à la Okamoto–Schnorr» nous prémunit contre la fabrication d'argent en raison de l'impossibilité de «falsifications supplémentaires» (voir chapitre 6). Quant à la technique utilisée dans le retrait simplifié de la figure 7.6, elle garantit l'interception d'un fraudeur par «double dépense».

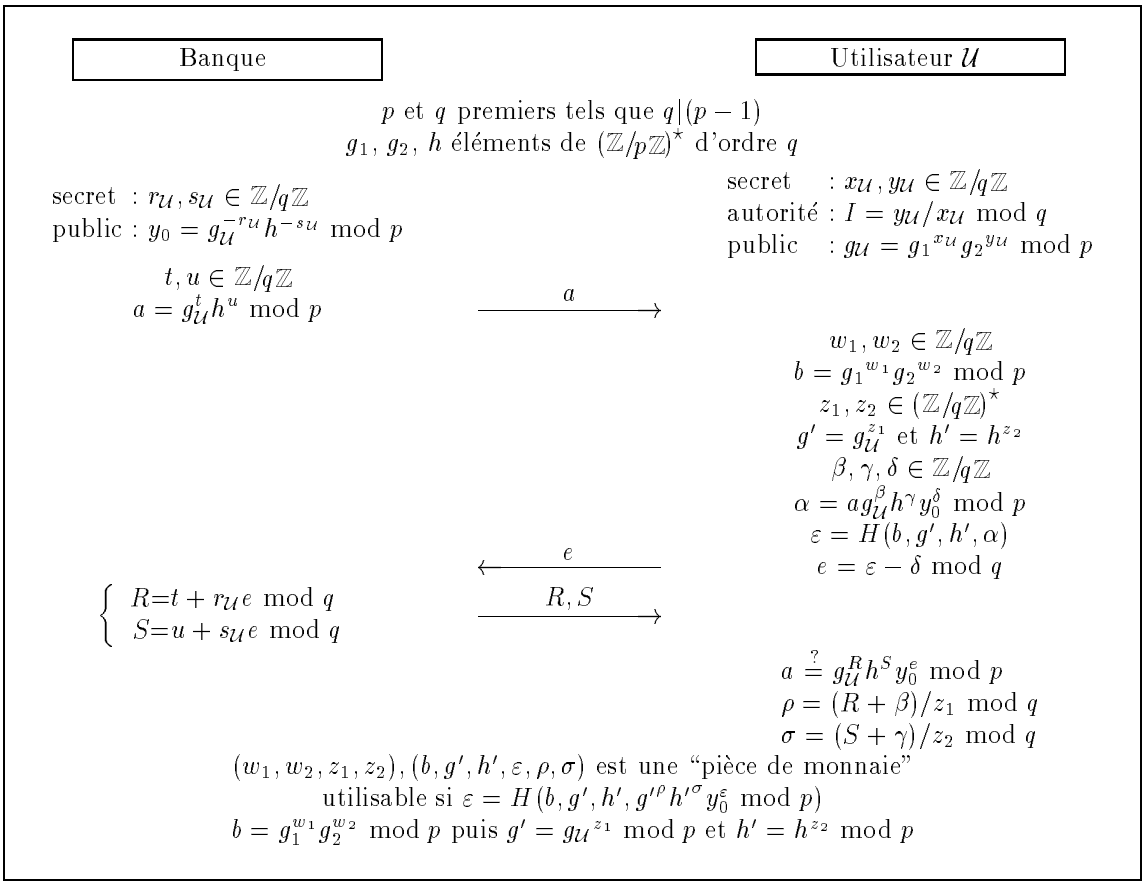

Fig. 7.8 – *Retrait*

La mise en place du protocole nécessite une autorité qui supervise toutes les Banques. Cette autorité choisit deux éléments g_1 et g_2 de $(\mathbb{Z}/p\mathbb{Z})^\star$ d'ordre q, a *priori* indépendants. Puis elle choisit aléatoirement A, B, C et D dans $(\mathbb{Z}/q\mathbb{Z})^\star$ et calcule $\Delta = AD - BC \bmod q$ ainsi que $h = g_1^A g_2^B \bmod p$ et $y_0 = g_1^C g_2^D \bmod p$. L'élément y_0 est publié comme clé publique de la Banque (nous ne considérons qu'une Banque, mais on peut bien entendu fabriquer autant de clés publiques que l'on veut de cette manière afin de faire participer plusieurs Banques).

Pour s'enregistrer, Alice entre en contact avec l'autorité qui choisit aléatoirement $x_\mathcal{A}$ et $y_\mathcal{A}$ dans $(\mathbb{Z}/q\mathbb{Z})^\star$, tels que $x_\mathcal{A} B - y_\mathcal{A} A \neq 0 \bmod q$, et calcule l'identifiant d'Alice, $I_\mathcal{A} = y_\mathcal{A}/x_\mathcal{A} \bmod p$ ainsi que sa clé publique, $g_\mathcal{A} = g_1^{x_A} g_2^{y_A} \bmod p$. L'identifiant $I_\mathcal{A}$ est gardé secret par l'autorité. Quant à la clé publique $g_\mathcal{A}$, elle est communiquée à tous, mais particulièrement à la Banque, elle, et en fait elle seule, en aura besoin lors d'un retrait. L'autorité calcule alors $\Delta_\mathcal{A} = I_\mathcal{A} A - B \bmod q$ et $\Gamma_\mathcal{A} = I_\mathcal{A} C - D \bmod q$ puis la clé secrète de la Banque associée à Alice, $r_\mathcal{A} = -1/x_\mathcal{A} \times \Delta/\Delta_\mathcal{A} \bmod q$ et $s_\mathcal{A} = -\Gamma_\mathcal{A}/\Delta_\mathcal{A} \bmod q$.

$$\begin{aligned} g_\mathcal{A}^{-r_\mathcal{A}} h^{-s_\mathcal{A}} &= g_1^{x_\mathcal{A}\Delta/x_\mathcal{A}\Delta_\mathcal{A}} g_2^{y_\mathcal{A}\Delta/x_\mathcal{A}\Delta_\mathcal{A}} g_1^{A\Gamma_\mathcal{A}/\Delta_\mathcal{A}} g_2^{B\Gamma_\mathcal{A}/\Delta_\mathcal{A}} \\ &= g_1^{((AD-BC)+A(I_\mathcal{A}C-D))/\Delta_\mathcal{A}} g_2^{(I_\mathcal{A}(AD-BC)+B(I_\mathcal{A}C-D))/\Delta_\mathcal{A}} \\ &= g_1^{C(AI_\mathcal{A}-B)/\Delta_\mathcal{A}} g_2^{D(AI_\mathcal{A}-B)/\Delta_\mathcal{A}} = g_1^C g_2^D \end{aligned}$$

$$g_{\mathcal{A}}^{-r_{\mathcal{A}}} h^{-s_{\mathcal{A}}} = y_0 \bmod p.$$

Un tel protocole (voir figure 7.8) ne révèle rien sur la clé secrète $(x_{\mathcal{A}}, y_{\mathcal{A}})$ d'Alice, et la partie publique de la pièce $(b, g', h', \varepsilon, \rho, \sigma)$ ne révèle rien sur l'identité d'Alice ni sur son identifiant $I_{\mathcal{A}}$.

Pour être en mesure de découvrir l'identité ou l'identifiant du fraudeur en cas de «double dépense», nous allons introduire la mise en gage initiale, de la preuve de connaissance d'une décomposition de g' relativement à g_1 et g_2 suivant le protocole d'Okamoto–Schnorr, dans le hachage de l'acquisition de la pièce.

7.3.4 Dépense d'une pièce

La dépense du certificat s'effectue par le contrôle de sa validité, puis de la connaissance d'une clé secrète associée, par les protocoles de Schnorr et d'Okamoto–Schnorr (voir figure 7.9). On remarque qu'en cas de «double dépense», avec probabilité écrasante, on

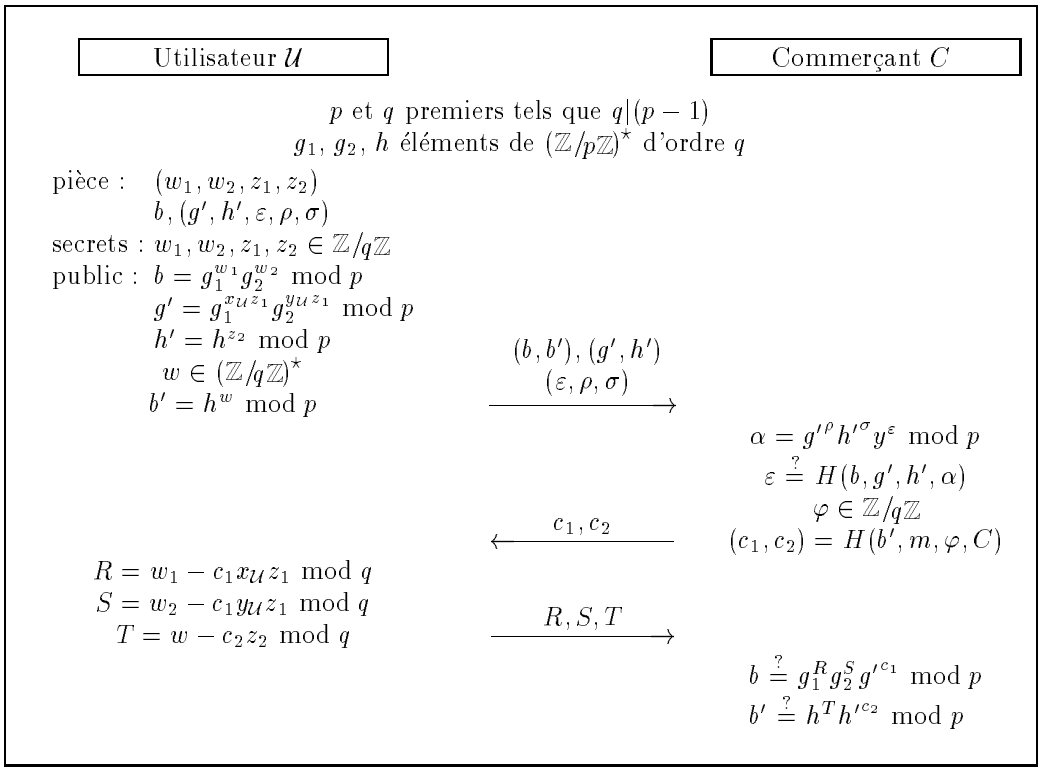

Fig. 7.9 – *Dépense*

obtient le couple (r', s') qui satisfait $g' = g_1^{r'} g_2^{s'} \bmod p$.

Alors, à moins d'une fraude, dont nous garantissons l'impossibilité, la pièce contient l'identifiant $I_{\mathcal{A}} = y_{\mathcal{A}}/x_{\mathcal{A}} = s'/r' \bmod q$. Par conséquent, une «double dépense» de cette pièce révèle l'identifiant du fraudeur.

7.3.5 Dépôt d'argent

Le commerçant C est alors en possession du certificat $(b, g', h', \varepsilon, \rho, \sigma)$ fourni par le client. Ce certificat satisfait l'égalité $\varepsilon = H(b, g', h', g'^{\rho} h'^{\sigma} y^{\varepsilon} \bmod p)$. Le commerçant possède également le résultat de la preuve de connaissance de décomposition de g' et h', vue comme une signature, (b', φ, R, S, T), d'un montant m : $b = g_1^R g_2^S g'^{c_1} \bmod p$ et $b' = h^T h'^{c_2} \bmod p$ avec $(c_1, c_2) = H(b', m, \varphi, C)$. Cette pièce utilisée peut se résumer au multiplet $(b', m, \varphi, C, R, S, T, g', h', \varepsilon, \rho, \sigma)$ vérifiant

$$\varepsilon = H(g_1^R g_2^S g'^{c_1} \bmod p, g', h', g'^{\rho} h'^{\sigma} y^{\varepsilon} \bmod p) \text{ et } b' = h^T h'^{c_2} \bmod p,$$

où $(c_1, c_2) = H(m, \varphi, C)$.

La Banque, après avoir vérifié la validité de toutes ces données, crédite le compte du commerçant C du montant m.

7.3.6 Sécurité

Trois notions de sécurité sont enfin présentes dans ce schéma. Tout d'abord, on va être sûr de retrouver un fraudeur qui effectue une «double dépense». Ensuite, il est normal que la Banque soit assurée du contrôle de la quantité d'argent en circulation. Le problème de la «double dépense» étant exclu, il ne reste qu'à garantir qu'après ℓ interactions avec la Banque, un utilisateur ne puisse détenir plus de ℓ pièces. Enfin, les utilisateurs ont également droit à un minimum de garanties. C'est-à-dire qu'il ne faut pas qu'ils puissent être accusés à tort de fraude. Notre protocole empêche la Banque elle-même de faire accuser un de ses clients.

7.3.6.1 Usurpation d'identité

Pour garantir la découverte d'un fraudeur en cas de «double dépense», il faut être sûr qu'il ne puisse posséder un coupon valide contenant une identité farfelue ou, pire, l'identité de quelqu'un d'autre.

Aucun des schémas de ce type ne peut, jusqu'à présent, prétendre une sécurité prouvée contre une usurpation d'identité selon une attaque parallèle d'un groupe (éventuellement important) de personnes. En effet, le schéma de Brands [8, 9] possédait le redoutable inconvénient de s'effondrer face à la coalition de plusieurs utilisateurs [13], mais également face un utilisateur isolé. La réparation de Schoenmakers [101], remodelée par Brands [11], fournit des schémas qui résistent aux attaques parallèles présentées pour le schéma initial. Cependant, aucune preuve ne garantit l'inexistence d'une attaque parallèle.

Pour notre schéma, nous pouvons affirmer le résultat suivant dont la preuve est identique à celle présentée au sujet du retrait simplifié.

Théorème 102 (Sécurité face à une personne).
Une usurpation d'identité, même selon une attaque parallèle, d'une seule personne est équivalente au logarithme discret.

Nous nous permettons d'émettre une unique conjecture pour terminer cette thèse, aucune démonstration ne permettant pour le moment de valider ce résultat.

Conjecture 103 (Sécurité face à plusieurs personnes).
Une usurpation d'identité, même selon une attaque parallèle, d'un groupe de personnes est équivalente au logarithme discret.

7.3.6.2 Contrôle de la quantité d'argent

Le contrôle du nombre de pièces en circulation est garanti par la sécurité de la signature en blanc sous-jacente contre les falsifications supplémentaires selon des attaques parallèles. Cette notion de falsification supplémentaire a d'ailleurs été définie dans ce simple but.

7.3.6.3 Garantie pour l'utilisateur

Nous pouvons désormais garantir le contrôle de l'argent par la Banque, mais quelle garantie ont les utilisateurs. Par exemple, dans les schémas de Brands et de Schoenmakers cités ci-dessus, la Banque connaît, et doit connaître la clé secrète de chaque utilisateur pour l'aider à fabriquer des pièces convenables. Mais avec cette connaissance, la Banque peut se fabriquer des pièces au nom d'Alice, et les dépenser plusieurs fois. Alice sera alors accusée, à tort, de «double dépense».

Dans notre schéma, la Banque n'a nullement besoin de connaître la clé secrète de chaque utilisateur. Une autorité extérieure lui fournit, une bonne fois pour toute, un couple $(r_{\mathcal{U}}, s_{\mathcal{U}})$ satisfaisant $y_0 = {g_{\mathcal{U}}}^{-r_{\mathcal{U}}} h^{-s_{\mathcal{U}}} \bmod p$. Même une puissance de calcul infinie ne permettra pas à la Banque de faire accuser Alice, car seule l'autorité possède l'identifiant $I_{\mathcal{A}}$ d'Alice.

Conclusion

Dans cette thèse, nous avons étudié la sécurité de divers schémas cryptographiques.

En effet, nous avons tout d'abord proposé deux nouveaux protocoles d'identification sûrs contre les attaques actives, à moins de connaître une méthode efficace pour résoudre PPP.

Ensuite, nous avons confirmé la sécurité des schémas de signature dérivés de protocoles d'identification à divulgation nulle de connaissance face à un vérifieur honnête tel que suggéré par Fiat et Shamir. Ainsi, les schémas de signature de Fiat-Shamir et de Schnorr n'admettent pas de falsification existentielle même face à des attaques à messages choisis adaptatives à moins de savoir résoudre, respectivement, le problème de la factorisation ou du logarithme discret dans les sous-groupes premiers. Puis nous avons utilisé cette nouvelle technique pour prouver la sécurité d'une légère variante du schéma de El Gamal (dont la version originale est existentiellement falsifiable selon des attaques sans message connu) sous réserve que le logarithme discret est difficile.

Avec des arguments semblables, mais des techniques plus fines, nous nous sommes intéressés aux schémas de signature en blanc. Nous avons alors présenté de tels schémas avec la preuve de l'impossibilité de falsifications supplémentaires même face à des attaques parallèles, à moins de savoir résoudre efficacement, respectivement, les problèmes du logarithme discret dans les sous-groupes premiers, RSA ou de la factorisation.

Nous avons enfin utilisé ces nouvelles signatures en blanc prouvées dans les schémas de Brands et de Schoenmakers afin de proposer un protocole de monnaie électronique prouvé sûr.

Toutefois, qu'il s'agisse des protocoles d'identification PPP, des schémas de signature électronique, des signatures en blanc ou même de monnaie électronique, les preuves de sécurité ont été données dans le modèle de l'oracle aléatoire.

Dans tous les cas, elle permettent de valider le «design» général des schémas proposés. Mais ne permettraient-elles pas de prouver des schémas pratiques ?

Cette thèse suggère donc deux problèmes ouverts essentiels :

1. est-il possible d'atteindre des résultats de sécurité exacte, comme présentés par Bellare et Rogaway [4], pour les schémas de signatures électronique et en blanc ?
   - le facteur Q ne pourrait-il pas être supprimé dans le temps des réductions ?
   - pour les signatures en blanc, la contrainte polylogarithmique sur le nombre d'interactions est vraiment très forte. Ne pourrait-on pas l'affaiblir ?
2. la notion abstraite d'oracle aléatoire peut-elle être rendue plus concrète ?

En fait, le deuxième problème est partiellement résolu :

- Pour ce qui est des schémas d'identification, Marc Girault et Jacques Stern [45] ont montré que la résistance aux collisions, ou aux «ℓ-collisions», de la fonction de mise en gage garantissait la sécurité d'un bon nombre de protocoles. Les résultats qu'ils énoncent pour PKP peuvent d'ailleurs être transcrits pour PPP.

- Récemment, Serge Vaudenay et moi-même [88] avons étudié une alternative pratique à l'oracle aléatoire dans le cadre des schémas de signature. En effet, en remplaçant l'oracle aléatoire par une boîte noire, contenant un élément d'une famille de fonctions pseudo-aléatoires, nous parvenons à prouver les mêmes résultats de sécurité.

  A priori, l'utilisation de fonctions pseudo-aléatoires conviendrait également pour les signatures en blanc et la monnaie électronique.

Glossaire

Asymétrique : Alice possède deux clés, une privée, une publique. Dans un schéma de chiffrement, la clé publique permet à Bob, ou à n'importe qui, de chiffrer un message pour Alice. Elle peut ensuite, et elle seulement, le déchiffrer à l'aide de sa clé privée. Dans un schéma de signature, la clé privée permet à Alice de signer un document. Tout le monde pourra vérifier cette signature à l'aide de la clé publique d'Alice.

Attaque : opération qui tente de violer un système. Dans cette thèse, on ne considèrera que les attaques des schémas de signature.

sans message connu : l'attaquant n'a à sa disposition que les données publiques du protocole.

à messages connus : l'attaquant a à sa disposition toute une liste de couples (message – signature) en plus des données publiques du protocole.

à messages choisis : l'attaquant a à sa disposition, en plus des données publiques du protocole, toute une liste de couples (message – signature) qu'il a choisis. Si l'attaque est

générique : ce choix est fait avant la vue de la clé publique.

orientée : ce choix est fait en fonction de la clé publique de l'individu dont on veut prendre l'identité.

adaptative ou dynamique : ce choix est fait tout au long de l'attaque.

Authentification : preuve d'identité en général. Cela regroupe l'identification et la signature.

Blum (module de) : module RSA $N = pq$ où p et q sont des entiers premiers congrus à 3 modulo 4. Dans l'anneau $(\mathbb{Z}/N\mathbb{Z})^\star$, la fonction carrée est une bijection de l'ensemble des résidus quadratiques dans lui-même et les racines carrées sont simples à calculer lorsqu'on connaît p et q.

Cassage (total) : découverte de la clé secrète d'un utilisateur.

Chiffrement : transformation appliquée à un message pour assurer sa *confidentialité*. Il peut être déterministe ou probabiliste. Dans ce dernier cas, seul le destinataire est capable de relier un message et son chiffré.

Clé : donnée qui personnalise un protocole cryptographique. Cette donnée peut être confidentielle, on parle dans ce cas de «clé secrète» ou privée. Elle peut également être publique, on parle alors de «clé publique».

Confidentialité : garantie que seules les personnes autorisées ont accès à l'information. Pour cela, on utilise des procédés de *chiffrement*.

Cryptanalyse : étude des procédés de *décryptage*, ou plus généralement de la sécurité des procédés cryptographiques.

Cryptographie : étude du *chiffrement* et du *déchiffrement*, ainsi que des procédés permettant d'assurer l'*intégrité*, l'*authentification* et la *signature*.

Cryptologie : étude scientifique de la *cryptographie* et de la *cryptanalyse*.

Déchiffrement : transformation inverse du *chiffrement* qui consiste à retrouver l'information initiale contenue dans le message chiffré à l'aide d'une clé secrète.

Décryptage : action de «casser» le *chiffrement* d'une information, et donc de retrouver le texte clair d'un chiffré, sans connaître la clé secrète.

Divulgation nulle de connaissance : traduction de «zero-knowledge» [48]. La preuve de connaissance ne laisse passer aucune information de plus que nécessaire, quelle que soit l'honnêteté du vérifieur.

face à un vérifieur honnête : dans ce cas, la preuve de connaissance ne laisse passer aucune information de plus que nécessaire lorsque le vérifieur pose les questions honnêtement, c'est-à-dire aléatoirement en général.

Falsification : fabrication d'un objet dont on n'a, «théoriquement», pas le pouvoir. Dans le cadre des schémas de signature électronique, les falsifications essentielles sont :

universelle : existence d'une machine capable de simuler le signeur «légal», sans sa clé secrète.

existentielle : possibilité de fabriquer un couple (message – signature) valide sans la connaissance de la clé secrète (le message n'est, en général, pas choisi par l'attaquant).

Pour ce qui est des signatures en blanc, la falsification à exclure est la falsification

supplémentaire : possibilité de fabriquer plus de couples (message – signature) que de signatures en blanc envoyées par l'autorité.

Fonction

à sens-unique : fonction simple à calculer mais difficile à inverser.

à sens-unique à trappe : fonction à sens-unique. Mais une information supplémentaire appelée «trappe» permet une inversion aisée.

de mise en gage : fonction permettant de s'engager sur une valeur sans la révéler.

négligeable : f est une fonction négligeable si pour tout c, il existe K tel que,

$$\forall k > K, f(k) < k^c.$$

polylogarithmique : fonction polynomiale en le logarithme de la variable. En général, cette variable sera la taille de la clé publique, donc le logarithme de cette clé.

Identification : preuve d'identité lors d'un contrôle d'accès.

Indistinguabilité : deux distributions de probabilité peuvent être plus ou moins proches et donc plus ou moins distinguables. Voici deux critères d'indistinguabilité.

polynomiale : on dit que deux distributions δ et δ' sont polynomialement indistinguables si pour tout distingueur $\mathcal{D}$, machine de Turing polynomiale qui tente de deviner la distribution suivie par son entrée, ne possède qu'une espérance de

distinction négligeable.

$$|E_\delta[\mathcal{D}(x)] - E_{\delta'}[\mathcal{D}(x)]| \ll 1/poly.$$

statistique : on dit que deux distributions δ et δ' sont statistiquement indistinguables si leur distance est négligeable.

$$\sum_y |\Pr_\delta[x = y] - \Pr_{\delta'}[x = y]| \ll 1/poly.$$

Intégrité : préserver un message contre des modifications volontaires ou non.

Multi-ensemble : ensemble où chaque élément peut apparaître plusieurs fois. On peut également voir cela comme un multiplet non ordonné.

Signature : donnée de faible taille jointe à un message qui prouve l'identité de l'auteur et qui garantit l'intégrité du message.

Symétrique : une seule clé est associée à chaque couple d'interlocuteurs. Cette unique clé sert à chiffrer et à déchiffrer.

Témoin : chaîne de caractères qui permet la vérification polynomiale de l'appartenance d'un mot à un langage.

- **caché** : traduction de «witness-hiding» [38]. Notion un peu plus faible que la *divulgation nulle de connaissance*. À l'issue de preuves de connaissance, il est impossible, même pour le vérifieur, de calculer un témoin.
- **indistinguable** : traduction de «witness indistinguishable» [38]. Toute preuve de connaissance utilise un témoin. Une preuve à témoin indistinguable est tout d'abord à témoin caché, mais de plus, quel que soit le témoin utilisé, la distribution du ruban de communication est la même.

Turing (machine de) : machine abstraite capable d'effectuer n'importe quel calcul. C'est une formalisation d'un «ordinateur». Une machine de Turing possède un ruban de travail sur lequel est initialement écrit une donnée x, un ruban aléatoire qui fournit une séquence de bits aléatoires nécessaires à son fonctionnement si cette machine est probabiliste, puis une fonction de changement d'état qui décrit le «calcul» à effectuer. Une telle machine de Turing est dite polynomiale s'il existe un polynôme P tel que le temps d'exécution est borné par $P(|x|)$, où $|x|$ est la longueur, en bits, de x.

Index

Bibliographie

[1] Arora S., Lund C., Motwani R., Sudan M. et Szegedy M. «Proof Verification and Hardness of Approximation Problems». In *Proceedings of the 33rd Symposium on the Foundations of Computer Science FOCS*, pages 14–23, Pittsburgh, Pennsylvanie, États-Unis, 1992. IEEE.

[2] Bellare M., Goldreich O. et Sudan M. «Free Bits, PCPs and Non-Approximability — Towards Tight Results». In *Proceedings of the 36th Symposium on the Foundations of Computer Science FOCS*, Milwaukee, Wisconsin, États-Unis, 1995. IEEE.
`http://www-cse.ucsd.edu/~mihir/papers/complexity-papers.html`.

[3] Bellare M. et Rogaway P. «Random Oracles are Practical: a paradigm for designing efficient protocols». In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 62–73, Fairfax, Virginie, États-Unis, 1993. ACM press.

[4] Bellare M. et Rogaway P. «The Exact Security of Digital Signatures – How to Sign with RSA and Rabin». In Maurer U., Ed., *Advances in Cryptology – Proceedings of EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416, Saragosse, Espagne, 1996. Springer-Verlag.

[5] Ben-Or M., Goldreich O., Goldwasser S., Håstad J., Kilian J., Micali S. et Rogaway P. «Everything Provable is Provable in Zero-Knowledge». In Goldwasser S., Ed., *Advances in Cryptology – Proceedings of CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, pages 37–56, Santa-Barbara, Californie, 1989. Springer-Verlag.

[6] Bleichenbacher D. «Generating El Gamal Signatures Without Knowing The Secret Key». In Maurer U., Ed., *Advances in Cryptology – Proceedings of EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 10–18, Saragosse, Espagne, 1996. Springer-Verlag.

[7] Boyar J., Chaum D., Damgård I. et Pedersen T. P. «Convertible Undeniable Signatures». In Menezes A. J. et Vanstone S. A., Eds., *Advances in Cryptology – Proceedings of CRYPTO '90*, volume 537 of *Lecture Notes in Computer Science*, Santa-Barbara, Californie, 1991. Springer-Verlag.

[8] Brands S. A. «An Efficient Off-line Electronic Cash System Based On The Representation Problem». Rapport technique, CWI (Centrum voor Wiskunde en Informatica), 1993.
`http://www.cwi.nl/cwi/publications` CS-R9323.

[9] Brands S. A. «Untraceable Off-line Cash in Wallets with Observers». In Stinson D. R., Ed., *Advances in Cryptology – proceedings of CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 302–318, Santa-Barbara, Californie, 1994. Springer-Verlag.

[10] Brands S. A. «A Note on Parallel Executions of Restrictive Blind Issuing Protocols for Secret-Key Certificates». Rapport technique, CWI (Centrum voor Wiskunde en Informatica), 1995.
`http://www.cwi.nl/cwi/publications` CS-R9519.

[11] Brands S. A. «More on Restrictive Blind Issuing of Secret-key Certificates in Parallel Mode». Rapport technique, CWI (Centrum voor Wiskunde en Informatica), 1995.
`http://www.cwi.nl/cwi/publications` CS-R9534.

[12] Brands S. A. «Off-Line Electronic Cash Based on Secret-Key Certificates». In *Proceedings of the 2nd International Symposium of Latin American Theoretical INformatics (LATIN' 95)*, Valparaíso, Chili, april 1995.
`http://www.cwi.nl/cwi/publications` CS-R9506.

[13] Brands S. A. «Restrictive Blind Issuing of Secret-key Certificates in Parallel Mode». Rapport technique, CWI (Centrum voor Wiskunde en Informatica), 1995.
`http://www.cwi.nl/cwi/publications` CS-R9523.

[14] Brands S. A. «Secret-Key Certificates». Rapport technique, CWI (Centrum voor Wiskunde en Informatica), 1995.
`http://www.cwi.nl/cwi/publications` CS-R9510.

[15] Burmester M. et Desmedt Y. «All Language in NP Have Divertible Zero-Knowledge Proofs and Arguments under Cryptographic Assumptions». In Damgard I. B., Ed., *Advances in Cryptology – Proceedings of EUROCRYPT '90*, volume 473 of *Lecture Notes in Computer Science*, pages 1–10, Aarhus, Danemark, 1991. Springer-Verlag.

[16] Camenisch J., Maurer U. et Stadler M. «Digital Payment Sysytems with Passive Anonymity-Revoking Trustees». In Bertino E., Kurth H., Martella G. et Montolivo E., Eds., *Proceedings of the 4th European Symposium On Research In Computer Security (ESORICS '96)*, volume 1146 of *Lecture Notes in Computer Science*, Rome, Italie, 1996. Springer-Verlag.

[17] Camenisch J., Piveteau J.-M. et Stadler M. «Fair Blind Signatures». In Guillou L. et Quisquater J.-J., Eds., *Advances in Cryptology – Proceedings of EUROCRYPT '95*, volume 921 of *Lecture Notes in Computer Science*, pages 209–219, Saint-Malo, France, 1995. Springer-Verlag.

[18] CAMENISCH J., PIVETEAU J.-M. et STADLER M. «An Efficient Fair Payment System». In *Proceedings of the 3rd ACM Conference on Computer and Communications Security*, pages 88–94, New Delhi, Inde, 1996. ACM press.

[19] CHAUM D. «Blind Signatures for Untraceable Payments». In Chaum D., Rivest R. L. et Sherman A. T., Eds., *Advances in Cryptology – Proceedings of CRYPTO '82*, pages 199–203. Plenum, NY, 1983.

[20] CHAUM D. «Security Without Identification: Transaction Systems to Make Big Brother Obsolete». *Communications of the ACM 28*, 10, octobre 1985.

[21] CHAUM D. «Privacy Protected Payments: Unconditional Payer And/Or Payee Untraceability». In *Smartcard 2000*. North Holland, 1988.

[22] CHAUM D. «Zero-Knowledge Undeniable Signatures». In Damgard I. B., Ed., *Advances in Cryptology – Proceedings of EUROCRYPT '90*, volume 473 of *Lecture Notes in Computer Science*, pages 458–464, Aarhus, Danemark, 1991. Springer-Verlag.

[23] CHAUM D. «Designated Confirmer Signatures». In Santis A. D., Ed., *Advances in Cryptology – Proceedings of EUROCRYPT '94*, volume 950 of *Lecture Notes in Computer Science*, Pérouse, Italie, 1995. Springer-Verlag.

[24] CHAUM D., DEN BOER B., VAN HEYST E., MJØLSNES S. et STEENBEEK A. «Efficient Off-line Electronic Checks». In Quisquater J.-J. et Vandewalle J., Eds., *Advances in Cryptology – Proceedings of EUROCRYPT '89*, volume 434 of *Lecture Notes in Computer Science*, pages 294–301, Houthalen, Belgique, 1990. Springer-Verlag.

[25] CHAUM D., FIAT A. et NAOR M. «Untraceable Electronic Cash». In Goldwasser S., Ed., *Advances in Cryptology – Proceedings of CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, pages 319–327, Santa-Barbara, Californie, 1989. Springer-Verlag.

[26] CHAUM D. et VAN ANTWERPEN H. «Undeniable Signatures». In Brassard G., Ed., *Advances in Cryptology – Proceedings of CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 212–216, Santa-Barbara, Californie, 1990. Springer-Verlag.

[27] CHEN L. *Witness Hiding Proofs and Applications*. Thèse, Aarhus University, août 1994.

[28] CHEN L., DAMGÅRD I. B. et PEDERSEN T. P. «Parallel Divertibility of Proofs of Knowledge». In Santis A. D., Ed., *Advances in Cryptology – Proceedings of EUROCRYPT '94*, volume 950 of *Lecture Notes in Computer Science*, pages 140–155, Pérouse, Italie, 1995. Springer-Verlag.

[29] DEN BOER B. et BOSSELAERS A. «An Attacks on the Last Two Rounds of MD4». In Feigenbaum J., Ed., *Advances in Cryptology – Proceedings of CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 194–203, Santa-Barbara, Californie, 1992. Springer-Verlag.

[30] DESMEDT Y., GOUTIER C. et BENGIO S. «Special Uses and Abuses of the Fiat-Shamir Passport Protocol». In Pomerance C., Ed., *Advances in Cryptology – Proceedings of CRYPTO '87*, volume 293 of *Lecture Notes in Computer Science*, pages 21–39, Santa-Barbara, Californie, 1988. Springer-Verlag.

[31] DIFFIE W. et HELLMAN M. E. «New Directions in Cryptography». In *IEEE Transactions on Information Theory*, volume IT–22, no. 6, pages 644–654, novembre 1976.

[32] DOBBERTIN H. «Cryptanalysis of MD5 Compress». Unpublished.

[33] DOBBERTIN H. «Cryptanalysis of MD4». In Gollmann D., Ed., *Third Fast Software Encryption*, volume 1039 of *Lecture Notes in Computer Science*, pages 53–69, Cambridge, Royaume Uni, 1996. Springer-Verlag.

[34] DOBBERTIN H. «The Status of MD5 After a Recent Attack». *CryptoBytes*, 2(2):1–6, été 1996.

[35] EL GAMAL T. «A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms». In *IEEE Transactions on Information Theory*, volume IT–31, no. 4, pages 469–472, juillet 1985.

[36] ESPEL LLIMA R. «Émulateur de Carte à Puce». Rapport technique, Laboratoire d'Informatique de l'École Normale Supérieure, june 1995. Rapport de stage.

[37] FEIGE U., FIAT A. et SHAMIR A. «Zero-Knowledge Proofs of Identity». *Journal of Cryptology*, 1:77–95, 1988.

[38] FEIGE U. et SHAMIR A. «Witness Indistinguishable and Witness Hiding Protocols». In *Proceedings of the 22nd ACM Symposium on the Theory of Computing STOC*, pages 416–426, Baltimore, Maryland, États-Unis, 1990. ACM Press.

[39] FERGUSON N. «Extensions of Single Term Coins». In Stinson D. R., Ed., *Advances in Cryptology – proceedings of CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 292–301, Santa-Barbara, Californie, 1994. Springer-Verlag.

[40] FERGUSON N. «Single Term Off-Line Coins». In Helleseth T., Ed., *Advances in Cryptology – Proceedings of EUROCRYPT '93*, volume 765 of *Lecture Notes in Computer Science*, pages 318–328, Lofthus, Norvège, 1994. Springer-Verlag.

[41] FIAT A. et SHAMIR A. «How to Prove Yourself: practical solutions of identification and signature problems». In Odlyzko A. M., Ed., *Advances in Cryptology – Proceedings of CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194, Santa-Barbara, Californie, 1987. Springer-Verlag.

[42] FRANKEL Y., TSIOUNIS Y. et YUNG M. «"Indirect Disclosure Proof": Achieving Efficient Fair Off-Line E-Cash». In Kim K. et Matsumoto T., Eds., *Advances in Cryptology – Proceedings of ASIACRYPT '96*, volume 1163 of *Lecture Notes in Computer Science*, pages 286–300, KyongJu, Corée du Sud, 1996. Springer-Verlag.

[43] GAREY M. R. et JOHNSON D. S. *Computers and Intractability, A Guide to the Theory of $\mathcal{NP}$-Completeness*. Freeman, New-York, 1979.

[44] GIRARD J.-Y. *La Machine de Turing*. Sources du savoir. Éditions du Seuil, mai 1995.

[45] GIRAULT M. et STERN J. «On the Length of Cryptographic Hash-Values used in Identification Schemes». In Desmedt Y. G., Ed., *Advances in Cryptology – proceedings of CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 202–215, Santa-Barbara, Californie, 1994. Springer-Verlag.

[46] GOLDREICH O., MICALI S. et WIGDERSON A. «Proofs That Yield Nothing but Their Validity and a Methodology of Cryptographic Protocol Design». In *Proceedings of the 27th Symposium on the Foundations of Computer Science FOCS*, pages 174–187, Toronto, Canada, 1986. IEEE.

[47] GOLDREICH O., MICALI S. et WIGDERSON A. «How to Prove All $\mathcal{NP}$ Statements in Zero-Knowledge and a Methodology of Cryptographic Protocol Design». In Odlyzko A. M., Ed., *Advances in Cryptology – Proceedings of CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 171–185, Santa-Barbara, Californie, 1987. Springer-Verlag.

[48] GOLDWASSER S., MICALI S. et RACKOFF C. «The Knowledge Complexity of Interactive Proof Systems». In *Proceedings of the 17th ACM Symposium on the Theory of Computing STOC*, pages 291–304, Providence, Rhode Island, États-Unis, 1985. ACM Press.

[49] GOLDWASSER S., MICALI S. et RACKOFF C. «The Knowledge Complexity of Interactive Proof Systems». *SIAM journal of computing*, 18(1):186–208, février 1989.

[50] GOLDWASSER S., MICALI S. et RIVEST R. «A Digital Signature Scheme Secure Against Adaptative Chosen-Message Attacks». *SIAM journal of computing*, 17(2):281–308, avril 1988.

[51] GUILLOU L. C. et QUISQUATER J.-J. «A Practical Zero-Knowledge Protocol Fitted to Security Microprocessor Minimizing Both Transmission and Memory». In Günter C. G., Ed., *Advances in Cryptology – Proceedings of EUROCRYPT '88*, volume 330 of *Lecture Notes in Computer Science*, pages 123–128, Davos, Suisse, 1988. Springer-Verlag.

[52] GUILLOU L. C. et QUISQUATER J.-J. «A "Paradoxal" Identity-Based Signature Scheme Resulting from Zero-Knowledge». In Goldwasser S., Ed., *Advances in*

Cryptology – Proceedings of CRYPTO '88, volume 403 of *Lecture Notes in Computer Science*, pages 216–231, Santa-Barbara, Californie, 1989. Springer-Verlag.

[53] HÅSTAD J. «Pseudo-Random Generators under Uniform Assumptions». In *Proceedings of the 22nd ACM Symposium on the Theory of Computing STOC*, Baltimore, Maryland, États-Unis, 1990. ACM Press.

[54] HODGES A. *Alan Turing ou l'Énigme de l'Intelligence*. Bibliothèque scientifique. Payot, Paris, 1988. traduction de «Alan Turing: The Enigma», Simon & Schuster Co., 1983.

[55] IMPAGLIAZZO I., LEVIN L. et LUBY M. «Pseudo-Random Generation from One-Way Functions». In *Proceedings of the 21st ACM Symposium on the Theory of Computing STOC*, pages 12–24, Seattle, Washington, États-Unis, 1989. ACM Press.

[56] ITOH T. et SAKURAI K. «On the Complexity of Constant Round ZKIP for Possession of Knowledge». *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E76-A(1), janvier 1993.

[57] ITOH T., SAKURAI K. et SHIZUYA H. «Any Language in IP has a divertible ZKIP». In Imai H., Rivest R. L. et Matsumoto T., Eds., *Advances in Cryptology – Proceedings of ASIACRYPT '91*, volume 739 of *Lecture Notes in Computer Science*, pages 382–397, Fujiyoshida, Japon, 1993. Springer-Verlag.

[58] JAKOBSSON M. et YUNG M. «Revocable and Versatile Electronic Money». In *Proceedings of the 3rd ACM Conference on Computer and Communications Security*, pages 76–87, New Delhi, Inde, 1996. ACM press.

[59] KAHN D. *La Guerre des Codes Secrets : des hiéroglyphes à l'ordinateur*. Inter-Editions, 1980. traduction de "The Codebreakers: The Story of Secret Writing", MacMillan Publiblishing Co., New York, NY, 1967.

[60] KALISKI B. S. «The MD2 Message-Digest Algorithm». RFC 1319, avril 1992.

[61] LAMPORT L. «Password Authentication with Insecure Communication». *Communications of the ACM 24*, 11:770–771, novembre 1981.

[62] LANDAU E. *Handbuch der Lehre von der Verteilung der Primzahlen*. Teubner, Berlin, 1909.

[63] LEVIN L. «One-Way Functions and Pseudorandom Generators». In *Proceedings of the 17th ACM Symposium on the Theory of Computing STOC*, pages 363–368, Providence, Rhode Island, États-Unis, 1985. ACM Press.

[64] LUBY M. et RACKOFF C. «How to Construct Pseudorandom Permutations from Pseudorandom Functions». *SIAM Journal of Computing*, 17(2):373–386, 1988.

[65] MCELIECE R. J. «A Public-Key Cryptosystem Based on Algebraic Coding Theory». *DSN progress report*, 42-4, 1978.

[66] M'Raïhi D. «Cost-Effective Payment Schemes with Privacy Regulation». In Kim K. et Matsumoto T., Eds., *Advances in Cryptology – Proceedings of ASIACRYPT '96*, volume 1163 of *Lecture Notes in Computer Science*, pages 266–275, KyongJu, Corée du Sud, 1996. Springer-Verlag.

[67] Naccache D. et von Solms S. «On Blind Signatures and Perfect Crimes». *Computers and Security*, 11:581–583, 1992.

[68] Naor M. «Bit Commitment Using Pseudo-Randomness». In Brassard G., Ed., *Advances in Cryptology – Proceedings of CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 123–132, Santa-Barbara, Californie, 1990. Springer-Verlag.

[69] NIST. «Digital Signature Standard (DSS)». Federal Information Processing Standards PUBlication XX, Draft, août 1991.

[70] NIST. «Secure Hash Standard (SHS)». Federal Information Processing Standards PUBlication YY, Draft, janvier 1992.

[71] NIST. «Secure Hash Standard (SHS)». Federal Information Processing Standards PUBlication 180, Draft, avril 1993.

[72] NIST. «Digital Signature Standard (DSS)». Federal Information Processing Standards PUBlication 186, novembre 1994.

[73] NIST. «Secure Hash Standard (SHS)». Federal Information Processing Standards PUBlication 180–1, avril 1995.

[74] Ohta K. et Okamoto T. «A Modification of the Fiat-Shamir Scheme». In Goldwasser S., Ed., *Advances in Cryptology – Proceedings of CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, pages 232–243, Santa-Barbara, Californie, 1989. Springer-Verlag.

[75] Ohta K. et Okamoto T. «Divertible Zero-Knowledge Interactive Proofs and Commutative Random Self-Reducibility». In Quisquater J.-J. et Vandewalle J., Eds., *Advances in Cryptology – Proceedings of EUROCRYPT '89*, volume 434 of *Lecture Notes in Computer Science*, pages 134–149, Houthalen, Belgique, 1990. Springer-Verlag.

[76] Ohta K. et Okamoto T. «Universal Electronic Cash». In Feigenbaum J., Ed., *Advances in Cryptology – Proceedings of CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 324–337, Santa-Barbara, Californie, 1992. Springer-Verlag.

[77] Okamoto T. «Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes». In Brickell E. F., Ed., *Advances in Cryptology – Proceedings of CRYPTO '92*, volume 740 of *Lecture Notes in Computer Science*, pages 31–53, Santa-Barbara, Californie, 1992. Springer-Verlag.

[78] Ong H. et Schnorr C. «Fast Signature Generation with a Fiat-Shamir-Like Scheme». In Damgard I. B., Ed., *Advances in Cryptology – Proceedings of EUROCRYPT '90*, volume 473 of *Lecture Notes in Computer Science*, pages 432–440, Aarhus, Danemark, 1991. Springer-Verlag.

[79] Ostrovsky R. et Wigderson A. «One-Way Functions are Essential for Non-Trivial Zero-Knowledge Proofs». (preliminary manuscript).

[80] Papadimitriou C. et Yannakakis M. «Optimization, Approximation, and Complexity Classes». *Journal of Computer and Systems Sciences*, 43:425–440, 1991.

[81] Patarin J. *Étude des Générateurs de Permutations Pseudo-aléatoires Basés sur le Schéma du DES*. Thèse, Université de Paris VI, novembre 1991.

[82] Pohlig S. C. et Hellman M. E. «An Improved Algorithm for Computing Logarithms over GF(p) and its Cryptographic Significance». *IEEE Transactions on Information Theory*, IT–24(1):106–110, janvier 1978.

[83] Pointcheval D. «A New Identification Scheme Based on The Perceptrons Problem». In Guillou L. et Quisquater J.-J., Eds., *Advances in Cryptology – Proceedings of EUROCRYPT '95*, volume 921 of *Lecture Notes in Computer Science*, pages 319–328, Saint-Malo, France, 1995. Springer-Verlag.

[84] Pointcheval D. «Les Réseaux de Neurones et leurs Applications Cryptographiques». Rapport technique, Laboratoire d'Informatique de l'École Normale Supérieure, février 1995.
`ftp://ftp.ens.fr/pub/reports/liens` LIENS-95-2.

[85] Pointcheval D. et Stern J. «Provably Secure Blind Signature Schemes». In Kim K. et Matsumoto T., Eds., *Advances in Cryptology – Proceedings of ASIACRYPT '96*, volume 1163 of *Lecture Notes in Computer Science*, pages 252–265, KyongJu, Corée du Sud, 1996. Springer-Verlag.

[86] Pointcheval D. et Stern J. «Security Proofs for Signature Schemes». In Maurer U., Ed., *Advances in Cryptology – Proceedings of EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 387–398, Saragosse, Espagne, 1996. Springer-Verlag.

[87] Pointcheval D. et Stern J. «New Blind Signatures Equivalent to Factorization». In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, Zurich, Suisse, 1997. ACM press.

[88] Pointcheval D. et Vaudenay S. «On Provable Security for Digital Signature Algorithms». Rapport technique, Laboratoire d'Informatique de l'École Normale Supérieure, octobre 1996.
`ftp://ftp.ens.fr/pub/reports/liens` LIENS-96-17.

[89] POUPARD G. «Implémentation sur carte à puce d'un Protocole d'Authentification». Rapport technique, Laboratoire d'Informatique de l'École Normale Supérieure, juillet 1995. Rapport de stage.

[90] POUPARD G. «Étude des Paramètres de deux Schémas d'Authentification et de Signature à Divulgation Nulle de Connaissance». Rapport technique, Laboratoire d'Informatique de l'École Normale Supérieure, juin 1996. Rapport de stage.

[91] PRENEEL B. *Analysis and Design of Cryptographic Hash Functions.* Thèse, Katholieke Universiteit Leuven, Departement Elektrotechniek, janvier 1993.

[92] RABIN M. O. «Digital Signatures». In Lipton R. et Millo R. D., Eds., *Foundations of secure computation.* Academic Press, New York, 1978.

[93] RABIN M. O. «Digitalized Signatures and Public Key Functions as Intractible as Factorization». Rapport technique, Massachusetts Institute of Technology – Laboratory for Computer Science, janvier 1979. MIT/LCS/TR-212.

[94] RIVEST R. «The MD4 Message-Digest Algorithm». In Menezes A. J. et Vanstone S. A., Eds., *Advances in Cryptology – Proceedings of CRYPTO '90*, volume 537 of *Lecture Notes in Computer Science*, pages 303–311, Santa-Barbara, Californie, 1991. Springer-Verlag.

[95] RIVEST R. «The MD4 Message-Digest Algorithm». RFC 1320, avril 1992.

[96] RIVEST R. «The MD5 Message-Digest Algorithm». RFC 1321, avril 1992.

[97] RIVEST R., SHAMIR A. et ADLEMAN L. «A Method for Obtaining Digital Signatures and Public Key Cryptosystems». *Communications of the ACM*, 21(2):120–126, février 1978.

[98] SCHNORR C. P. «Efficient Identification and Signatures for Smart Cards». In Brassard G., Ed., *Advances in Cryptology – Proceedings of CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 235–251, Santa-Barbara, Californie, 1990. Springer-Verlag.

[99] SCHNORR C. P. «Efficient Signature Generation by Smart Cards». *Journal of Cryptology*, 4(3):161–174, 1991.

[100] SCHNORR C. P. «Security of 2^t-Root Identification and Signatures». In Koblitz N., Ed., *Advances in Cryptology – proceedings of CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 143–156, Santa-Barbara, Californie, 1996. Springer-Verlag.

[101] SCHOENMAKERS L. A. M. «An Efficient Electronic Payment System Withstanding Parallel Attacks». Rapport technique, CWI (Centrum voor Wiskunde en Informatica), 1995.
`http://www.cwi.nl/cwi/publications` CS-R9522.

[102] SHAMIR A. «An Efficient Identification Scheme Based on Permuted Kernels». In Brassard G., Ed., *Advances in Cryptology – Proceedings of CRYPTO '89*, volume 435 of *Lecture Notes in Computer Science*, pages 606–609, Santa-Barbara, Californie, 1990. Springer-Verlag.

[103] SHAMIR A. «$\mathcal{IP} = \mathcal{PSPACE}$». *Journal of the ACM*, 39(4):869–877, octobre 1992.

[104] SHANKS D. «Class number, a theory of factorization, and genera». In *Proceedings of the symposium on Pure Mathematics*, volume 20, pages 415–440. AMS, 1971.

[105] SHANNON C. E. «A Mathematical Theory of Communication». *Bell system technical journal*, 27(4):379–423, 623–656, 1948.

[106] SHANNON C. E. «Communication Theory of Secrecy Systems». *Bell system technical journal*, 28(4):656–715, 1949.

[107] SHOR P. W. «Algorithms for Quantum Computation: Discrete Logarithms and Factoring». In *Proceedings of the 35th Symposium on the Foundations of Computer Science FOCS*, pages 124–134, Santa Fe, Nouveau Mexique, États-Unis, 1994. IEEE.

[108] SHOR P. W. «Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer». *SIAM Journal of Computing*, 1996.

[109] SHOUP V. «On The Security of a Practical Identification Scheme». In Maurer U., Ed., *Advances in Cryptology – Proceedings of EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 344–353, Saragosse, Espagne, 1996. Springer-Verlag.

[110] SKUBISZEWSKI M. *Optimisation par Recuit Simulé : mise en œuvre matérielle de la machine de Boltzmann, application à l'étude des suites synchronisantes.* Thèse, Université d'Orsay, juin 1993.

[111] SMITH J. L. «The Design of Lucifer, A Cryptographic Device for Data Communications». Rapport technique, IBM, 1971. RC3326.

[112] STERN J. «A New Identification Scheme Based on Syndrome Decoding». In Stinson D. R., Ed., *Advances in Cryptology – proceedings of CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 13–21, Santa-Barbara, Californie, 1994. Springer-Verlag.

[113] STERN J. «Designing Identification Schemes with Keys of Short Size». In Desmedt Y. G., Ed., *Advances in Cryptology – proceedings of CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 164–173, Santa-Barbara, Californie, 1994. Springer-Verlag.

[114] STERN J. «The Validation of Cryptographic Algorithms». In Kim K. et Matsumoto T., Eds., *Advances in Cryptology – Proceedings of ASIACRYPT '96*, volume 1163 of *Lecture Notes in Computer Science*, pages 301–310, KyongJu, Corée du Sud, 1996. Springer-Verlag.

[115] TURING A. «On Computable Numbers with an Application to the *Entscheidungsproblem*». In *Proceedings of the Mathematical Society*, volume 42, pages 230–265, 1936–37. Errata dans le volume 43, pp544-546, 1937.

[116] U.S. NATIONAL BUREAU OF STANDARD. *Data Encryption Standard*, 1977.

[117] VAN HEYST E. et PEDERSEN T. P. «How to Make Fail-Stop Signatures». In Rueppel R. A., Ed., *Advances in Cryptology – Proceedings of EUROCRYPT '92*, volume 658 of *Lecture Notes in Computer Science*, pages 366–377, Balatonfüred, Hongrie, 1993. Springer-Verlag.

[118] VAUDENAY S. «Hidden Collisions on DSS». In Koblitz N., Ed., *Advances in Cryptology – proceedings of CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 83–88, Santa-Barbara, Californie, 1996. Springer-Verlag.

[119] VERNAM G. S. «Cipher Printing Telegraph Systems for Secret Wire and Radio Telegraphic Communications». *Journal of the American Institute of Electrical Engineers*, 45:109–115, 1926.

[120] YANNAKAKIS M. «On The Approximation of Maximum Satisfiability». In *Proceedings of the 33rd Symposium on the Foundations of Computer Science FOCS*, Pittsburgh, Pennsylvanie, États-Unis, 1992. IEEE.

[121] YAO A. C. «Theory and Applications of Trapdoor Functions». In *Proceedings of the 23rd Symposium on the Foundations of Computer Science FOCS*, pages 80–91, Chicago, Illinois, États-Unis, 1982. IEEE.