UNIVERSITÉ de CAEN BASSE-NORMANDIE

U.F.R. de Sciences

ÉCOLE DOCTORALE SIMEM

La Cryptographie au Service de la Protection de la Vie Privée

Mémoire présenté et soutenu publiquement le 2 décembre 2009 à l'Université de Caen Basse-Normandie

en vue de l'obtention du

Diplôme d'Habilitation à Diriger des Recherches de l'Université de Caen Basse-Normandie

par

Sébastien Canard
Orange Labs

MEMBRES du JURY

Directrice de recherche :	
Brigitte Vallée	Directrice de Recherche au CNRS, laboratoire GREYC (Caen)
Rapporteurs :	
Marc Joye	Habilité à Diriger des Recherches, Ingénieur à Thomson R&D
Olivier Pereira	Professeur à l'UCL, Microelectronics Laboratory
David Pointcheval	Directeur de Recherche au CNRS, laboratoire LIENS (ENS)
Examinateurs :	
Marc Girault	Habilité à Diriger des Recherches, Professeur au lycée Victor Hugo
Gilles Zémor	Professeur des Universités, Université de Bordeaux II, laboratoire IMB

à Aline et à mes enfants,
Louis, Eléonore, et Gautier

Préface

L'année 2009 a été très mouvementée : la naissance de mon fils Gautier, la crise économique, le jour où Louis et moi avons enfin battu le comte Doku à Légo Star Wars, la grippe A, le départ de mon mentor pour d'autres estrades, la première année d'école d'Eléonore, Federer qui gagne enfin Roland Garros... Cette Habilitation à Diriger des Recherches n'a pas dérogé à la règle et il a fallu le concours de nombreuses personnes pour que je puisse, enfin, rédiger ces quelques lignes. Je vais bien évidemment me servir de cette préface pour nommer chacune d'elles, mais un peu plus loin.

La recherche m'a happé en 1999 et ne m'a depuis plus lâché. Pourtant, elle a parfois été tentée : des (bons !) articles refusés, des collaborations qui disparaissent, des difficultés pour effectuer ses recherches de façon optimale... Malgré tout, je m'épanouis dans mon travail parce qu'il m'apporte tellement de belles choses : l'adrénaline qui monte quand on sent que l'on touche une idée (géniale), le plaisir de voir dans sa boîte de messagerie "We are pleased to announced you...", le regard lumineux de son thésard lorsqu'il vous expose sa dernière idée (elle aussi géniale), les applaudissements de son auditoire après la présentation de son dernier résultat, les discussions animées avec des collaborateurs que l'on apprécie au delà du métier de chercheur...

Mon domaine de recherche est celui de la cryptographie. Celui-ci a le mérite de pouvoir être abordée de différentes manières. De la manipulation d'outils mathématiques très complexes à la construction d'un schéma en "légos cryptographiques", en passant par l'étude de la complexité en moyenne d'un algorithme, il y a dans ce domaine de multiples façon d'innover, et de multiples personnes aux compétences diverses avec qui échanger.

Je ne sais pas trop qui je dois remercier en premier, parmi toutes les personnes qui ont, d'une façon ou d'une autre, contribué à cette habilitation. Je ne suis pas certain qu'il existe une quelconque convention sur ce point et je vais donc le faire dans un ordre quelconque.

Iwen et Amandine, vous avez essuyé les plâtres de mon encadrement de doctorants et je tiens à vous en remercier. Si vous n'en êtes peut-être pas conscients, vous m'avez énormément apporté et je vous dois en grande partie cette habilitation. Je tiens par ailleurs à m'excuser car vous avez certainement été les cobayes de ma maîtrise parfois hasardeuse de la direction d'un thésard. Je souhaite aussi prévenir Roch qu'il va certainement continuer à être l'objet de mes expériences pendant les trois prochaines années et je le remercie d'avoir accepté mon encadrement.

L'une des difficultés des remerciements est qu'il faut chercher à faire des regroupements. De ce point de vue, Marc est inclassable car il appartient à beaucoup de catégories : il m'a enseigné la cryptographie, il m'a appris le métier de chercheur, il a été mon directeur de thèse,

il a signé des articles avec moi, il a relu plusieurs de mes articles sans en être auteur, il a aussi relu ce mémoire alors qu'il était rempli de fautes d'orthographe (je sais qu'il n'aime pas du tout ça), il est dans mon jury aujourd'hui et il est aussi mon ami. Je tiens particulièrement à le remercier de faire partie de mon jury, sachant que son nouveau métier lui laisse moins de libertés.

Merci Brigitte d'avoir accepté d'être ma Directrice de Recherche pour cette habilitation. Je regrette de ne pas avoir plus souvent l'occasion de profiter de ton immense expérience et de tes connaissances hors paires. J'espère en avoir plus souvent l'occasion dans le futur.

Après un petit décompte, j'arrive à la conclusion que j'ai eu jusqu'à ce jour 21 co-auteurs différents. La palme revient tout naturellement à Jacques (avec 11 papiers communs) et Aline[1] (qui se retrouve avec 7 collaborations fructueuses depuis 2006). Cela a été un bonheur de travailler avec vous et j'espère pouvoir continuer encore longtemps. Je vous remercie aussi d'avoir passé du temps à améliorer ce mémoire. Jacques, j'ai beaucoup appris auprès de toi et je sais que je dois encore t'observer travailler car tu as encore des choses à m'enseigner, notamment ta façon remarquable et très diplomatique de montrer les faiblesses que peuvent avoir les constructions des autres (notamment les miennes...). Aline, ta rigueur, ta volonté et tes choix de titres nous ont permis d'obtenir plusieurs résultats dont je suis particulièrement fier. Je sais que ta nouvelle position à CryptoExperts t'accorde moins de temps mais j'espère que nous aurons l'occasion de retravailler ensemble, avant que tu n'ouvres ta pizzeria ou que ta carrière à la "Star'Ac" débute.

En faisant mon décompte, je me suis aussi rendu compte que j'étais presque parvenu à publier un article avec l'ensemble de mes co-bureaux successifs (Nizar, il faut qu'on se fasse prochainement une réunion pour remédier à ce petit manque). Benjamin est arrivé un jour en ayant une idée extraordinaire d'une carte de fidélité universelle sécurisée où il fallait utiliser des signatures de groupe. Nous avons finalement publié un article et, même si celui-ci ne parle finalement pas de signatures de groupe, je suis très heureux d'avoir cette contribution avec toi. Je souhaite aussi remercier Hervé S., dont la curiosité et l'impressionnante capacité à trouver des idées a forcément donné naissance à plusieurs contributions, dont certaines qui sont encore dans nos cartons et qui mériteraient un jour de ressortir. Emeline a aussi partagé quelques mois mon bureau et nous avons écrit ensemble trois articles. Merci pour ta bonne humeur et nos discussions autour des cocottes et des canards.

Même si je n'ai pas partagé de bureau avec eux, je souhaite aussi remercier mes autres co-auteurs qui ont travaillé dans les mêmes locaux que moi : Benoît, Cécile (Poupoule pour compléter la basse-cour...), Eric, Fabien, Fabrice, Matthieu, Michel, Pascal et Stéphane. Je pourrais certainement dire un mot sur chacun de vous mais j'ai peur de devenir un peu long et ennuyeux. Une petite ligne toutefois pour Fabrice dont la disponibilité et la compréhension m'ont guidé dans de nombreuses occasions. Une autre pour Fabien qui est pour moi un ami et que je remercie pour le temps qu'il a consacré pour cette habilitation. Il y a aussi ceux avec qui je n'ai pas écrit d'article mais avec qui de multiples discussions ont forcément agrémenté ceux que je possède : Elvis, David A., David L., Gildas, Gilles, Henri, Hervé D., Matt, Olivier, Thomas, Yannick.

Il me reste encore à remercier Berry, Damien, Jonathan et Martijn pour leur collabo-

1. Pour éviter toute confusion, je tiens à préciser ici qu'il existe dans mon entourage deux Aline. L'une, Aline G., avec qui je travaille et l'autre, Aline C., qui partage ma vie. Je pense que chacune se reconnaîtra tout au long de ce mémoire, et c'est bien là le principal.

ration dans les articles que j'ai en commun avec eux. Il y a aussi tous les stagiaires que j'ai encadrés et qui ont apporté leur contribution à ce mémoire : Ahmad, Aymen, Bienvenu, Céline, Christophe, Déborah, Jean-Charles, Mohamed et Vanessa. Et puis les personnes avec qui j'ai travaillé par le biais des projets collaboratifs CRYPTO++, SAVE et PACE, et notamment Christophe, Eric, Georg, Jean-Luc, Malika, Moez, Olivier, Pascal, Paul, Pierre-Alain, Thomas, Valérie...

Pour passer une Habilitation à Diriger des Recherches il faut aussi faire un certain nombre de démarches administratives. En cela, je souhaite remercier Mme Ygouf, pour sa rigueur et sa gentillesse et Régine (ainsi que les personnes qui travaillent avec elle) pour son soutien.

J'en arrive maintenant à mes rapporteurs. Marc, tu n'avais pas pu être rapporteur de ma thèse, à quelques jours près, et je suis maintenant très content que ce soit réparé. Merci pour ton travail et ta rigueur sur cette habilitation, et pour ta gentillesse dans la vie. David, je te suis très reconnaissant pour avoir accepté d'être rapporteur, pour toutes les discussions que nous avons, notamment dans le cadre des projets CRYPTO++, SAVE et PACE. Je remercie aussi Olivier d'avoir montré de l'intérêt pour mes travaux en acceptant de commenter ce mémoire, qui plus est dans un délai aussi court. Enfin, merci à Gilles d'avoir accepté d'être dans mon jury. Même si nos activités cryptographiques sont assez différentes, je suis très content de travailler avec toi depuis quelques temps.

Je vais finir par les personnes qui me sont les plus chères. Aline, les soirées et les week-ends de ces derniers mois ont été consacrés en grande partie à la rédaction de ce mémoire, alors que tu avais certainement d'autres idées en tête. Merci pour ton soutien et pour les milles et une autres choses qui font que la vie à tes côtés est ce qu'il y a de plus beau. Enfin, merci à mes trois enfants, Louis, Eléonore et Gautier, qui me font redécouvrir toutes ces petites choses de la vie que l'on a tendance à oublier en grandissant.

Avant-propos

La cryptographie est étymologiquement la science du secret et existe depuis de nombreux siècles. D'abord utilisée pour garantir la confidentialité des données, elle s'est depuis démocratisée en assurant la sécurité des services de télécommunications, étendant de ce fait son champ d'action à l'authentification d'une personne ou d'un message, la non-répudiation, l'intégrité mais aussi l'anonymat des transactions.

Cet anonymat est parfois primordial dans les nouveaux services de télécommunications et entre aujourd'hui dans un domaine plus vaste dénommé en anglais "privacy" et le plus souvent traduit par "protection de la vie privée". Ce domaine consiste à offrir aux utilisateurs de services un maximum de garanties sur la non-divulgation de leurs données personnelles. Les travaux de recherche que je vais décrire dans ce mémoire s'inscrivent dans une démarche d'élargissement du domaine de la cryptographie à la protection de la vie privée.

Il existe de nombreux outils cryptographiques, parmi lesquels les signatures de groupe ou les signatures aveugles, qui seront abordées dans ce mémoire, permettant aux utilisateurs d'être anonymes. Une grand part de la recherche que j'ai menée ces dernières années consiste en l'utilisation et l'amélioration de ces outils. Pour autant, comme le montrent les travaux que j'ai menés, il est possible de relâcher les contraintes dues à l'anonymat, et on peut protéger sa vie privée sans pour autant être anonyme.

Ce mémoire présente mes travaux de recherche dans le domaine de la protection de la vie privée. Le premier chapitre va introduire les outils cryptographiques que j'ai étudiés. Le Chapitre 2 explicite l'utilisation de ces briques cryptographiques dans différents services pour lesquels la protection de la vie privée est préconisée. Le cas particulier de la monnaie électronique est abordé dans le Chapitre 3. Enfin, le dernier chapitre se focalise sur la cryptographie "(très) bas-coût" (aussi appelée cryptographie "(ultra) légère") pour la protection de la vie privée.

Table des matières

Chapitre 1

Contributions aux Outils Cryptographiques

Mes travaux de recherche portent sur le domaine de la cryptographie. Ce chapitre est consacré à la présentation de la cryptographie ainsi que des principales briques cryptographiques nécessaires à la compréhension de ce mémoire. Je vais m'attacher plus particulièrement à présenter les constructions d'outils cryptographiques auxquelles j'ai contribué. Je vais présenter, ou simplement aborder, mes résultats sur les signatures de groupe et leurs variantes [CT03a, CSST06], sur les preuves de connaissance [CSST06, CD07, CCT07], les signatures déléguées [CLM08, CJ09] ou la gestion de clés dans un groupe hiérarchique [CJ08]. Ceux-ci ont été obtenus en collaboration avec des chercheurs aussi bien du laboratoire de sécurité de Orange Labs dans lequel je travaille que de la communauté scientifique extérieure, ou bien par le biais de l'encadrement de mes deux doctorants : Amandine Jambert [CJ08, CJ09] et Iwen Coisel [CCT07].

Sommaire

Les objectifs de la cryptographie moderne sont multiples et, parmi les principaux, on peut citer la confidentialité, l'authentification (de personnes ou de messages avec la signature électronique), l'intégrité ou bien l'anonymat. J'ai étudié une partie significative des briques cryptographiques permettant d'atteindre ces objectifs. Comme je l'ai dit précédemment, il serait long et inutile de décrire tous les outils cryptographiques que j'ai dû utiliser depuis mes débuts en recherche. Je vais donc uniquement m'intéresser à ceux qui sont pour moi les plus importants.

Je vais tout d'abord aborder les preuves de connaissance à divulgation nulle de connaissance. Initialement introduites pour des besoins en authentification, elles sont aujourd'hui utilisées dans des domaines beaucoup plus vastes, comme notamment l'anonymat. Pour cela, elles deviennent parfois difficiles à utiliser, d'où l'importance de les étudier pour les rendre moins complexes.

Dans le domaine de la signature électronique, je me suis à nouveau beaucoup plus intéressé aux schémas n'offrant pas uniquement la certitude sur la provenance du message et l'intégrité de ce dernier. Dans ma recherche, il me faut ainsi parfois cacher cette signature (à l'aide de preuves de connaissance), cacher l'identité du signataire, permettre à d'autres de modifier une partie du message ou bien obtenir des signatures sur des messages ayant eux-mêmes des propriétés spécifiques.

Pour ce qui est de la confidentialité, je vais à nouveau sortir du schéma classique de chiffrement/déchiffrement. En effet, je vais souvent avoir besoin de prouver qu'un message, ayant par ailleurs des propriétés spécifiques, est correctement chiffré. Ou bien ce sont les clés cryptographiques utilisées pour chiffrer qui vont avoir certaines propriétés spécifiques, comme par exemple celle d'appartenir à un groupe hiérarchique.

Je vais m'efforcer dans ce chapitre de donner un aperçu des briques que j'ai utilisées, ou auxquelles j'ai contribué par ma recherche.

1.1 Les Preuves de Connaissance

Les preuves de connaissance à divulgation nulle de connaissance permettent à une entité de prouver que certaines données secrètes, appelées dans la suite plus simplement "secrets", $(\mathsf{sk}_1, \ldots, \mathsf{sk}_r)$ dont elle possède la connaissance vérifient une relation $\mathcal{R}$ bien précise connue du vérifieur. Ces preuves sont par la suite notées

$$\textsc{Pok}(\mathsf{sk}_1, \ldots, \mathsf{sk}_r : \mathcal{R}(\mathsf{sk}_1, \ldots, \mathsf{sk}_r)).$$

Dans les constructions qui m'intéressent, et qui vont alimenter les pages de ce mémoire, les secrets en question sont des logarithmes discrets de valeurs connues. Nous avons aujourd'hui à notre disposition les résultats suivants :

- preuve de connaissance d'un logarithme discret [Sch89, GPS06]: $\textsc{Pok}(\alpha : y = g^\alpha)$;
- preuve de connaissance d'une représentation [DF02]: $\textsc{Pok}(\alpha_1, \ldots, \alpha_q : y = g_1^{\alpha_1} \cdots g_q^{\alpha_q})$;
- preuve d'égalité de logarithmes discrets [CP92]: $\textsc{Pok}(\alpha : y = g^\alpha \wedge z = h^\alpha)$;

- preuve qu'une valeur engagée appartient à un intervalle I [Bou00, Lip03, Gro05, CCS08]: $\text{POK}(\alpha : y = g^\alpha \wedge \alpha \in I)$;
- preuve de connaissance d'un double logarithme discret [Sta96, NS00]: $\text{POK}(\alpha : z = g^\alpha \wedge y = g_1^{g_2^\alpha})$;
- preuve de connaissance d'un secret parmi n, ou preuve du "ou" [CDS94, SCPY94]: $\text{POK}(\alpha : T_1 = g^{h_1^\alpha} \vee y = g^{h_2^\alpha})$.

A noter que toutes ces preuves de connaissance sont interactives et peuvent être transformées en signatures (de connaissance), à l'aide de l'heuristique de Fiat-Shamir [FS86]. Cette construction, notée SOK par la suite, a été prouvée sûre, dans le modèle de l'oracle aléatoire, par Pointcheval et Stern [PS00].

J'ai apporté diverses contributions à ces preuves de connaissance, soit en proposant des constructions particulières pour les preuves d'appartenance à des intervalles, soit en généralisant la preuve d'égalité de logarithmes discrets, soit en prouvant la sécurité d'une construction générique.

1.1.1 Preuve d'Appartenance à un Intervalle

Il existe deux familles de preuve d'appartenance d'un secret à un intervalle. Dans la première, la preuve n'est pas exacte et donne finalement uniquement la certitude que le secret appartient à un intervalle "un tout petit peu plus grand" que celui initialement espéré. En pratique, cela n'est pas nécessairement problématique lorsque l'intervalle est très grand et si le secret a une forme ou une construction particulière. C'est par exemple le cas dans le schéma de signature de groupe ACJT [ACJT00].

Par contre, lorsqu'il s'agit de prouver qu'on a moins de 25 ans sans révéler son âge, il est bien évidemment nécessaire que la preuve utilisée soit exacte. Il existe alors trois grandes familles de méthode :

1. celle qui utilise certaines propriétés des nombres positifs. Ainsi, pour prouver que $x \in [a,b]$, je vais prouver que $x-a \geq 0$ et que $b-x \geq 0$. Boudot [Bou00], puis Lipmaa [Lip03] et Groth [Gro05] ont alors utilisé le fait qu'un nombre positif se décompose en somme de trois ou quatre carrés : $X = x_1^2 + x_2^2 + x_3^2 + x_4^2$. Il reste alors à faire une preuve de connaissance à divulgation nulle de connaissance que les valeurs $x - a$ et $b - x$ se décomposent en la somme de quatre éléments qui sont par ailleurs des carrés ;
2. celle que Camenisch *et al.* [CCS08] ont plus récemment proposé, notamment lorsque l'intervalle est petit, de publier une signature de type Camenisch-Lysyanskaya de chaque élément de l'intervalle $[a,b]$. Ainsi, prouver que son secret $x \in [a,b]$ consiste à retrouver la signature Camenisch-Lysyanskaya qui a été publiée sur x (en effet, celle-ci existe si et seulement si x appartient à l'intervalle), puis à prouver qu'on connaît une signature sur x, sans révéler laquelle. Cette méthode a pour avantage d'être très efficace mais nécessite des paramètres publics très importants. Ils ont par ailleurs proposé une variante pour des intervalles plus grands ;
3. celle basée sur la décomposition binaire qui consiste à décomposer le secret x en binaire pour ensuite prouver son appartenance à l'intervalle. La preuve de connaissance complète consiste alors à prouver que la décomposition binaire est la bonne, en utilisant des preuves du "ou". Il est dans une premier nécessaire de comparer a et x bit à bit,

sans rien révéler sur le secret x, et donc en listant toutes les possibilités :

- $(x_{l-1} = 1$ et $a_{l-1} = 0)$ ou ;
- $(x_{l-1} = a_{l-1}$ et $x_{l-2} = 1$ et $a_{l-2} = 0)$ ou ;
- ... ;
- $(x_{l-1} = a_{l-1}$ et $x_{l-2} = a_{l-2}$ et ... et $x_1 = a_1$ et $x_0 = 1)$.

Cela revient donc à parcourir l'arbre des possibilités sur les bits du secret x. Pour des raisons évidentes, le cas le plus simple ici est lorsque a est une puissance de 2. Il faut ensuite faire la même chose pour b.

Avec Aline Gouget et Emeline Hufschmitt [CGH06], nous avons repris l'idée de la décomposition binaire pour prouver qu'un élément secret x est plus petit ou égal qu'un autre élément secret X. Le principe est très proche de celui exposé ci-dessus et je ne vais pas rentrer dans les détails.

Avec Céline Dulong, lors de son stage sous mon encadrement, nous nous sommes intéressés à la méthode de la décomposition binaire, en partant du principe que pour des petits secrets, celle-ci apparaît comme particulièrement intéressante [CD07]. Pourtant, telle que décrite ci-dessus, la preuve ne prend pas en compte le fait que le vérifieur connaît les valeurs de a et de b. Ainsi, si par exemple $a_{l-1} = 1$, il n'est pas nécessaire de s'intéresser à la première ligne ci-dessus puisqu'elle ne sera jamais vérifiée. Nous avons donc proposé un algorithme permettant d'optimiser au mieux les éléments dont il faut prouver la validité.

Ainsi, avec notre méthode, la preuve que le secret x appartient par exemple à l'intervalle $[a,b]$, avec $a = 8$ et $b = 13$, est alors la suivante.

$$\begin{gathered}\textsc{Pok}(\alpha,\beta,\gamma_0,\ldots,\gamma_3,\delta : (C_3/g = h^{\gamma_3}) \wedge \\ \Big(C_2 = h^{\gamma_2} \vee \big(C_2/g = h^{\gamma_2} \wedge (C_1 = h^{\gamma_1} \vee (C_1/g = h^{\gamma_1} \wedge C_0 = h^{\gamma_0}))\big)\Big) \wedge \\ C = g^{\alpha}h^{\beta} \wedge C\tilde{C}^{-1} = h^{\delta} \wedge \\ (C_1 = h^{\gamma_1} \vee C_1/g = h^{\gamma_1}) \wedge (C_0 = h^{\gamma_0} \vee C_0/g = h^{\gamma_0}))\end{gathered}$$

Dans le cas d'un groupe d'ordre premier de 1024 bits, avec des exposants de 160 bits, nous obtenons ainsi une méthode qui est en moyenne plus efficace que celle de Boudot lorsque le secret a une taille inférieure à 60 bits.

1.1.2 Généralisation de la Preuve d'Egalité

Dans [CSST06], j'ai proposé avec Berry Schoenmakers, Martijn Stam et Jacques Traoré une preuve de connaissance à divulgation nulle de connaissance qu'un ensemble de logarithmes discrets forme un sous-ensemble d'un autre ensemble de logarithmes discrets. Soit $(y_1,\ldots,y_n,a_1,\ldots,a_m) \in \mathbb{G}_q^{n+m}$, $1 \leq m \leq n$ un $(n+m)$-uplet tel que

$$\{\log_f a_j \mid 1 \leq j \leq m\} \subseteq \{\log_g y_i \mid 1 \leq i \leq n\}.$$

Si $n = m = 1$, il s'agit alors de la preuve d'égalité de logarithmes discrets due à Chaum et Pedersen [CP92]. Dans le cas plus général, nous avons dans un premier temps prouvé que si le logarithme discret de f en base g est inconnu, alors l'équation ci-dessus est vérifiée si les

témoins u_i, v_j, w_j vérifient les relations suivantes

$$\begin{aligned} y_i &= g^{u_i}, \text{ pour } i \in [1,n] \\ a_j &= f^{v_j}, \text{ pour } j \in [1,m] \\ \exists i \in [1,n] \text{ tel que } a_j y_i &= (fg)^{w_j}, \text{ pour } j \in [1,m] \end{aligned}$$

Notre preuve de connaissance consiste alors à prouver sa connaissance des témoins u_i, v_j, w_j vérifiant les trois équations ci-dessus, ce qui est relativement aisé et peu coûteux si n et m sont relativement petits, comme ce sera le cas dans notre utilisation [CSST06].

1.1.3 Sécurité des DLRS

Comme nous allons le voir dans la suite de ce mémoire, il est nécessaire d'avoir à sa disposition des preuves de connaissance à divulgation nulle de connaissance pour créer de nombreuses briques cryptographiques. Les secrets dont il faut prouver la connaissance vérifient parfois des équations relativement complexes qu'il est nécessaire d'étudier en détail afin de prouver la sécurité du système complet. Il semble ainsi intéressant d'avoir à sa disposition un outil permettant de se passer de la description de ces preuves et de la fourniture des preuves de sécurité.

Un première étape a été franchie par Kiayias, Tsiounis et Yung [KTY04] qui ont proposé une construction générique de toute preuve de connaissance portant sur un ensemble de relations de logarithmes discrets (DLRS pour Discrete Logarithm Relation Set), c'est-à-dire une combinaison des preuves de connaissance de logarithme discret, de représentation et d'égalité de logarithmes. Plus précisément, un ensemble de z relations de logarithmes discrets avec r variables libres $\mathsf{sk}_1,\ldots,\mathsf{sk}_r$, des objets $A_{1,1},\ldots,A_{r,1},\ldots,A_{r,z},B_1,\ldots,B_z$ à valeur dans un groupe $\mathbb{G}$ doit vérifier

- pour tout $w \in [1,r]$, $\mathsf{sk}_w \in]2^{l_w} - 2^{\mu_w}, 2^{l_w} + 2^{\mu_w}[$, où l_w et μ_w sont des paramètres de sécurité ;
- pour tout $i \in [1,z]$, la i-ème relation vérifie

$$B_i = \prod_{w=1}^{r} A_{w,i}^{\mathsf{sk}_w}.$$

La preuve de connaissance est alors définie par

$$\begin{aligned} \textsc{Pok}(\mathsf{sk}_1,\ldots,\mathsf{sk}_r \quad : \quad & B_1 = \prod_{w=1}^{r} A_{w,1}^{\mathsf{sk}_w} \wedge \cdots \wedge B_z = \prod_{w=1}^{r} A_{w,z}^{\mathsf{sk}_w} \\ & sk_1 \in]2^{l_1} - 2^{\epsilon(\mu_1+k)+2}, 2^{l_1} + 2^{\epsilon(\mu_1+k)+2}[\wedge \cdots \wedge \\ & sk_w \in]2^{l_w} - 2^{\epsilon(\mu_w+k)+2}, 2^{l_w} + 2^{\epsilon(\mu_w+k)+2}[\end{aligned}$$

où ϵ et k sont deux paramètres de sécurité. Il s'agit là d'une preuve inexacte d'appartenance à un intervalle.

Kiayias, Tsiounis et Yung [KTY04] ont montré que leur construction, présentée dans la Figure 1.1, est sûre, c'est-à-dire qu'elle vérifie les propriétés de consistance, de validité et de divulgation nulle de connaissance face à un vérifieur honnête, dans le cas où le groupe de

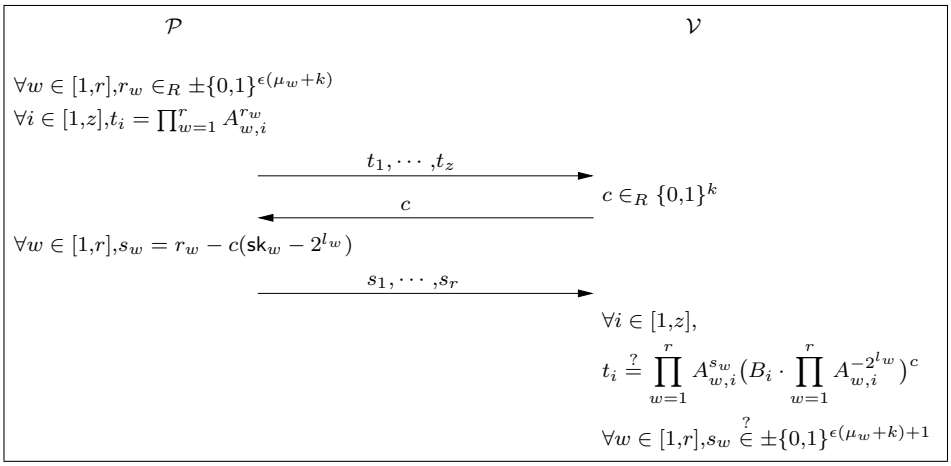

FIG. 1.1 – *Protocole pour une DLRS*

travail est le groupe $QR(n)$ des résidus quadratiques modulo un nombre composé n. Ils se sont cependant restreints aux DLRS triangulaires. Une DLRS est dite triangulaire si pour toute relation i contenant $b+1$ variables libres, les b premières forment un sous ensemble de l'union de toutes les variables libres présentes dans les $i-1$ précédentes relations. C'est par exemple le cas du schéma de signature de groupe ACJT [ACJT00] où il est possible d'ordonner les relations de la façon suivante :

- $T_2 = g^w$ où w est un nouveau secret ;
- $T_3 = g^e h^w$ où e est nouveau et w secret connu ;
- $1 = T_2^e / g^{ew}$ où ew est nouveau et e connu ;
- $a_0 = T_1^e / (a^x y^{ew})$ où x est nouveau et e et ew sont connus.

Pourtant, il existe d'autres cas d'utilisation de DLRS non triangulaires, comme par exemple les constructions présentées dans [DF02, CL02a]. Avec Iwen Coisel et Jacques Traoré [CCT07], j'ai proposé une preuve de sécurité de la construction ci-dessus qui n'impose pas de contrainte sur la DLRS utilisée. Plus précisément, nous nous sommes intéressés à la preuve de validité, puisque les preuves de consistance et de divulgation nulle de connaissance ne nécessitaient pas de modification par rapport à celles proposées dans [KTY04]. Cette preuve est relativement longue et complexe à présenter et je propose donc au lecteur intéressé de regarder [CCT07] ou [Coi09] pour les détails.

Notre preuve se restreint aussi au groupe $QR(n)$, qui est le plus difficile à appréhender et le plus couramment utilisé. En effet, le cas des groupes d'ordre premier est bien étudié et présente peu de difficultés depuis par exemple les travaux de Brands. A noter que très récemment, Camenisch *et al.* [CKY09] ont proposé le même genre de preuve dans le cas d'un groupe dit "protégé" qui inclut $QR(n)$.

1.2 Signatures Camenisch-Lysyanskaya

Le lien entre les signatures de groupe et les signatures Camenisch-Lysyanskaya (du nom de leurs inventeurs) est assez paradoxal. En effet, c'est à partir du schéma de signature de groupe

d'Ateniese *et al.* [ACJT00] que Camenisch et Lysyanskaya ont imaginé ce concept [CL02b]. Pour autant, c'est aujourd'hui à partir des signatures Camenisch-Lysyanskaya qu'il est possible de construire des schémas de signature de groupe (voir Section 1.3 ci-dessous).

Un schéma de signature Camenisch-Lysyanskaya [CL02b] est un schéma de signature classique, c'est-à-dire qu'il est constitué d'un algorithme KeyGen de génération des clés, d'une procédure de signature Sign et d'une autre de vérification Verif. A cela sont rajoutées diverses procédures qui sont les suivantes :

- un algorithme BlockSign qui permet au signataire de signer un message m divisé en plusieurs blocs $m_1 \| \ldots \| m_k$, chaque block étant traité de manière a priori indépendante ;
- une procédure CLSign qui prend en entrée la clé secrète de signature et un engagement sur plusieurs messages $(m_1 \ldots, m_k)$ et qui donne en sortie la signature du message $m = m_1 \| \ldots \| m_k$. A noter que le signataire n'a ici pas nécessairement la connaissance des blocs de message m_i ;
- un protocole entre un utilisateur et un signataire qui permet au premier d'obtenir du second la signature d'un message divisé en blocs, de telle sorte que le signataire n'a connaissance que d'une partie des blocs, alors que l'utilisateur en connaît l'ensemble. Une variante permet de faire en sorte qu'un ou plusieurs blocs soit calculés conjointement par le signataire et l'utilisateur, et inconnus au final du signataire ;
- une preuve de connaissance à divulgation nulle de connaissance qui permet à un utilisateur de prouver qu'il connaît une signature sur des blocs de message, sans révéler la signature ni tout ou une partie des blocs de message :

$$\text{Pok}(\sigma, m_1, \ldots, m_k : \sigma = \text{Sign}(m_1 \| \ldots \| m_k)).$$

Dans [CJ10b], Amandine Jambert et moi-même avons introduit un nouveau protocole qui permet à un utilisateur ayant préalablement obtenu une signature Camenisch-Lysyanskaya auprès du signataire, d'interagir à nouveau avec ce dernier pour rajouter des blocs de messages, certains éventuellement inconnus du signataire, sans pour autant révéler les blocs précédemment signés.

Il existe aujourd'hui plusieurs instances de schéma de signature Camenisch-Lysyanskaya. Certaines d'entre elles [CL02b, CL04, ASM07] sont résumées ci-dessous. Il en existe aussi d'autres qui ne sont pas détaillées [ASM08, Fuc09].

- [CL02b]
  - Clé secrète : (p,q) t.q. $n = pq$
  - Eléments publics : $b, a_0, \ldots, a_k \in QR(n)$
  - Signature : (A,e,s) tel que $A^e = a_0 b^s \prod_{i=1}^{k} a_i^{m_i} \pmod{n}$
  - Hypothèse : RSA Flexible [FO97]
- [CL04]
  - Clé secrète : $\gamma \in \mathbb{Z}_p$ où p est un nombre premier
  - Eléments publics : $h, h_0, \ldots, h_k \in \mathbb{G}_1$ où $\mathbb{G}_1$ est un groupe d'ordre premier noté p
  - Signature : (A,x,y) tel que $A^{\gamma+x} = h_0 h^y \prod_{i=1}^{k} h_i^{m_i} \pmod{p}$
  - Hypothèse : q-SDH [BB08]
- [ASM07]
  - Clé secrète : $X, x \in \mathbb{G} \times \mathbb{Z}_p^*$ où p est premier et $\mathbb{G}$ est un groupe d'ordre premier

- Eléments publics : $G_0, \ldots, G_3, H_1, \ldots, H_k, \tilde{H} \in \mathbb{G}^{k+4} \times \tilde{\mathbb{G}}$ où $\tilde{\mathbb{G}}$ est un groupe d'ordre p
- Signature : (A,x,y) tel que $\Sigma_1 = X(m_1 G_0^a)^c$, $\Sigma_2 = \left(G_1 G_2^a G_3^b H_1^{m_2} \cdots H_k^{m_k}\right)^{\frac{1}{x+c}}$ et $\tilde{\Sigma}_3 = \tilde{H}^c$
- Hypothèse : AWSM [ASM07]

1.3 Signatures de Groupe et Variantes

Les schémas de signature de groupe permettent aux membres du groupe de signer des messages anonymement au nom du groupe. L'appartenance au groupe est gérée par une entité souvent nommée manageur du groupe. Les signatures produites sont par ailleurs intraçables. Pourtant, le manageur d'ouverture, a priori différent du manageur de groupe, est capable de lever l'anonymat d'une signature donnée en entrée. Il devra alors faire une preuve que c'est bien le membre qu'il désigne qui est le coupable (on parle de preuve de "culpabilité").

1.3.1 Modélisation

Selon le modèle de sécurité BSZ [BSZ05], un schéma de signature de groupe doit vérifier les quatre propriétés de sécurité suivantes :

- la **consistance** qui spécifie que les signatures produites par un membre du groupe doivent toujours être acceptées et que ses signatures doivent toujours pouvoir être ouvertes par le manageur d'ouverture ;
- la **validité** stipule qu'un adversaire est incapable de produire une signature de groupe valide qui par ailleurs va entraîner une erreur à l'ouverture de cette signature ou une preuve de culpabilité qui est fausse ;
- l'**intraçabilité** doit montrer qu'un adversaire, après avoir choisi deux membres du groupe, est incapable de savoir lequel des deux a signé un message donné. Dans cette expérience, l'adversaire a le pouvoir du manageur du groupe et est même capable de corrompre les clés des deux membres ;
- la propriété de **non culpabilité** garantit que le manageur du groupe, le manageur d'ouverture et des membres corrompus doivent être incapables d'accuser faussement un membre du groupe honnête d'avoir produit une signature de groupe, si ce n'est pas le cas.

1.3.2 Une Construction Générique

Comme je l'ai déjà dit dans la Section 1.2, il est possible de construire un schéma de signature de groupe à partir d'un schéma de signature de type Camenisch-Lysyanskaya et d'un schéma de chiffrement. Il existe déjà plusieurs schémas de signature de groupe construits à partir de ce concept, comme par exemple [DP06] mais aussi [ACJT00, BBS04].

- GGen : cette phase de génération des clés va exécuter celles du schéma de signature Camenisch-Lysyanskaya (pour l'appartenance au groupe) et du schéma de chiffrement (pour la levée d'anonymat).

- GJoin : ce protocole est basé[1] sur le protocole du schéma de signature Camenisch-Lysyanskaya entre un utilisateur et un signataire qui permet au premier d'obtenir du second la signature Camenisch-Lysyanskaya σ d'un secret usk uniquement connu du membre du groupe et calculé conjointement par l'utilisateur et le manageur du groupe (ceci pour résister aux coalitions [ACJT00]). Comme nous avons pu le voir dans la Section 1.2, cette signature σ est composée d'un ou plusieurs éléments du groupe $(A_1,\ldots,A_\ell)$ et/ou un ou plusieurs exposants $(e_1,\ldots,e_k)$.
- GSig : cette procédure est exécutée par un membre du groupe qui va prendre en entrée un message m. Le membre va dans un premier temps chiffrer l'élément A_1, par exemple, de la signature σ, c'est-à-dire $C = \textsc{Enc}(A_1)$. Il va ensuite produire une signature de connaissance, en utilisant l'heuristique de Fiat-Shamir (nous nous plaçons donc ici dans le modèle de l'oracle aléatoire [BR93]),

  $$U = \textsc{Sok}(A_1,\ldots,A_\ell,e_1,\ldots,e_k,x : (A_1,\ldots,A_\ell,e_1,\ldots,e_k) = \textsc{Sign}(x) \wedge C = \textsc{Enc}(A_1))(m).$$

  La signature de groupe est finalement le couple (C,U).
- GVer : cette phase consiste simplement à vérifier la signature de connaissance U.
- GOpen : le manageur d'ouverture va ici prendre en entrée sa clé secrète de déchiffrement pour déchiffrer C afin de récupérer la valeur A_1. Celle-ci est alors utilisée pour faire le lien avec la procédure GJoin afin de retrouver l'identité du membre. Le manageur d'ouverture va finalement prouver qu'il a correctement déchiffré la valeur C en prouvant sa connaissance de la clé secrète de déchiffrement, utilisée pour déchiffrer C et contenue dans la clé publique.

Il est possible de montrer que le schéma de signature de groupe ainsi obtenu est sûr, dans le modèle de l'oracle aléatoire, sous l'hypothèse que le schéma de chiffrement a la propriété d'indistinguabilité[2] et que le schéma de signature Camenisch-Lysyanskaya est infalsifiable.

Il existe depuis peu d'autres constructions qui ne nécessitent pas l'oracle aléatoire. Celles-ci prennent pour base le papier de Groth et Sahai [GS08] sur les preuves non-interactives à témoins indistinguables pour les environnements bilinéaires. Initialement, le principe est très proche de la construction ci-dessus, si ce n'est que les preuves Groth-Sahai ne sont pas des preuves de connaissance mais des preuves d'appartenance. Il est donc nécessaire de rajouter un moyen de prouver sa connaissance des secrets sous-jacents à cette preuve. Par exemple dans [Gro07], ceci est fait en utilisant une signature jetable qui est certifiée à l'aide d'une clé secrète long terme dont la consistance est prouvée dans la preuve Groth-Sahai.

1.3.3 Une Variante avec Traçabilité Partielle

Lors de ma thèse, j'avais introduit avec Jacques Traoré le concept de signatures de liste [CT03a], permettant, contrairement aux signatures de groupe, de partiellement tracer les signatures produites par un même membre. Plus précisément, nous avions séparé le temps en plusieurs périodes pendant lesquelles il était possible pour tout un chacun de savoir si deux signatures de liste valides étaient produites par le même membre ou non, sans pour

1. Il existe des variantes plus ou moins complexes en fonction des propriétés que l'on recherche. Ainsi, dans [DP06], Delerablée et Pointcheval ont obtenu la sécurité du schéma dans un mode concurrent.
2. IND-CCA ou IND-CPA selon le niveau d'anonymat (fort ou faible) désiré.

autant savoir lequel. Il était par contre infaisable de relier deux signatures émises lors de deux périodes différentes.

A partir du schéma de signature de groupe générique présenté ci-dessus, il est relativement aisé de construire un schéma de signature de liste. Pour cela, il faut rajouter le calcul de la valeur $T = g_s^{\mathsf{usk}}$ où g_s est un élément propre à la période. Il convient aussi de rajouter que la valeur usk en exposant est bien la même que celle qui est signée dans σ. Il est clair que pour une séquence et un membre donnés, cette valeur T est toujours la même : ce membre est donc tracé. On montre par ailleurs, sous l'hypothèse DDH, que pour deux périodes différentes, et donc deux valeurs h_{c_1} et h_{c_2} aléatoires, il n'est pas possible de faire le lien entre les deux valeurs T_1 et T_2 correspondantes.

Dans [CSST06], j'ai travaillé avec Berry Schoenmakers, Martijn Stam et Jacques Traoré à une construction plus adaptée aux petits groupes. Chaque membre du groupe possède ainsi un secret x et deux valeurs publiques $y = g^x$ et z aléatoire. La clé publique du groupe est alors l'ensemble $(y_j,z_j)_{j=1,\ldots,N}$ s'il y a N membres dans le groupe (N petit). Un membre souhaitant produire une signature de liste lors de la période s va ainsi calculer $s = \mathcal{H}(T,1)$, $t = \mathcal{H}(T,0)$, et $X = \mathcal{H}'(m,T)$, où $\mathcal{H}$ et $\mathcal{H}'$ sont des fonctions de hachage. La signature de liste est alors $T_1 = t^{x_i}$, $T_2 = s^{x_i} {z_i}^X$ et une preuve de connaissance de x tel que $T_1 = t^x$, puis une preuve d'égalité de 1 parmi N que $\log_{gt}(yT_1)$ et $\log_s(T_2 z^{-X})$, telle que décrite dans la Section 1.1. Dans le cas de groupes avec peu de membres, il est possible de montrer que cette solution est plus efficace que la précédente.

Le même concept de traçabilité partielle et les mêmes techniques de construction ont depuis été utilisés dans [BCC04] pour mettre en place les DAA (Attestation Anonyme Directe) et dans [KTY04] pour les "signatures traçables". Pour les DAA, ce n'est pas le temps qui permet de tracer les utilisateurs mais les fournisseurs de service. En effet, dans certains cas, on peut souhaiter être anonyme vis-à-vis d'un fournisseur de services mais en étant par contre reconnu par ce dernier, afin d'assurer un suivi de session permettant d'obtenir des services adaptés. Ainsi, dans le concept des DAA, un membre du groupe va être traçable par un fournisseur de services alors que deux fournisseurs de service différents ne vont pas être capables de croiser leurs informations pour tracer un même utilisateur membre du groupe.

1.3.4 Les Signatures d'Anneau

Une autre variante bien connue des signatures de groupe sont les signatures d'anneau, introduites dans [RST01]. Celles-ci permettent en effet de signer des messages anonymes et intraçables au nom du groupe. En revanche, contrairement aux signatures de groupe, il n'existe pas d'autorité d'ouverture permettant de lever l'anonymat d'une signature. Par ailleurs, il n'existe pas non plus de manageur de groupe gérant les entrées et les sorties des membres au sein du groupe.

Plus précisément, le principe général est le suivant. Chacun peut s'il le souhaite publier des clés cryptographiques le concernant. Par la suite, lorsque quelqu'un ayant effectué cette phase préliminaire souhaite produire une signature d'anneau, il va lui-même composer son groupe en choisissant un certain nombre de clés publiques parmi celles proposées. Ainsi, une personne ne sait pas forcément qu'elle appartient à un groupe, à moins qu'on le lui dise, ou qu'elle ne dispose de la signature d'anneau en question.

Il existe plusieurs constructions de signatures d'anneau et la plupart d'entre elles n'offrent malheureusement que des solutions où la signature est linéaire en la taille de l'anneau, que ce soit en espace ou en temps de calcul. La seule construction n'ayant pas, à ce jour, cet inconvénient est celle proposée par Chandran, Groth et Sahai [CGS07]. En effet, dans ce système, la représentation de l'anneau par une matrice permet de rendre la signature plus compacte et sa génération moins coûteuse.

1.4 Les Signatures Aveugles

Une autre brique cryptographique bien connue permettant d'être anonyme est la signature aveugle, également appelée signature en blanc. Le principe est le suivant : un utilisateur interagit avec un signataire dans le but d'obtenir une signature sur un message de son choix inconnu du signataire. Ce dernier est par ailleurs incapable de reconnaître la signature (et a fortiori le message) qu'il vient de produire si on la lui présente. Ainsi, l'utilisateur est anonyme parmi l'ensemble des utilisateurs ayant demandé une telle signature à ce signataire.

Je me suis en fait le plus souvent intéressé à des variantes de ces signatures aveugles, où le signataire garde un certain contrôle sur ce qu'il est en train de signer. Dans le cas de la signature partiellement aveugle, le signataire conserve la connaissance d'une partie du message qu'il est en train de signer. Pour les signatures aveugles à anonymat révocable, une autorité désignée est habilitée à lever l'anonymat d'une signature de deux façons différentes : soit en fournissant le message et la signature obtenus à partir du protocole initial entre le signataire et l'utilisateur, soit en retrouvant l'identité de l'utilisateur lorsqu'il divulgue le message et la signature.

1.4.1 Signatures Aveugles à Anonymat Révocable

Les signatures aveugles à anonymat révocable ont été introduites par Camenisch, Piveteau et Stadler [SPC95] et ont plus récemment été formalisées par Hufschmitt et Traoré [HT07]. De façon très informelle, un tel schéma doit vérifier les propriétés suivantes :

- la **consistance** montre qu'un utilisateur ayant légitimement interagit avec le signataire sera nécessairement accepté lorsqu'il dévoilera le message et la signature obtenus. Par ailleurs, les deux procédures de levée d'anonymat doivent nécessairement fonctionner ;
- la **non falsification** est l'impossibilité pour un adversaire ayant interagi ℓ fois avec le signataire de fournir soit $\ell + 1$ signatures valides, soit des signatures dont les levées d'anonymat ne seront pas cohérentes, à la manière des signatures de groupe ;
- l'**intraçabilité** garantit, à l'instar des signatures de groupe, l'infaisabilité pour un adversaire de faire le lien entre un message et une signature donnés et deux protocoles d'obtention de signatures aveugles auxquels il a participé en tant que signataire ;
- la propriété de **non culpabilité** décrit le fait qu'un adversaire ne peut pas accuser faussement un utilisateur d'avoir obtenu de façon aveugle une signature et un message donnés.

J'ai proposé un schéma de signature aveugle à anonymat révocable [CGT04]. Cette construction a été la base de celle proposée par la suite par Hufschmitt et Traoré [HT07], avec une

instanciation différente et des propriétés de sécurité moins fortes[3]. En effet, dans les deux cas, il est possible de décrire le schéma à partir de n'importe quel schéma de signature Camenisch-Lysyanskaya. Dans [CGT04], nous avions pris l'instanciation basée sur le RSA flexible, alors que dans [HT07], le choix a été fait de prendre celle basée sur l'hypothèse q-SDH. De façon très informelle, nous avons les étapes suivantes :

- le protocole d'obtention d'une signature aveugle correspond au protocole d'obtention d'une signature Camenisch-Lysyanskaya où le message est seulement connu par l'utilisateur. La signature Camenisch-Lysyanskaya σ porte à la fois sur le message mais aussi sur un élément s uniquement connu par l'utilisateur et une clé secrète u propre à ce dernier. Celle-ci est insérée dans le message à signer par le biais de la valeur g^u envoyée par l'utilisateur au signataire. Il est par ailleurs nécessaire d'ajouter à cela, pour des raisons de levée d'anonymat, le chiffrement, pas l'utilisateur, de l'élément secret s ;
- lorsque l'utilisateur souhaite diffuser le message et la signature, il doit dans un premier temps chiffrer g^u puis effectuer une signature de connaissance de m, u, s et de la signature σ tels que cette dernière porte sur m, s et u et que c'est bien le même u qui est chiffré (par le biais de g^u), sans révéler u ni σ. Par contre, le message m est bien évidemment révélé, ainsi que l'élément s, pour les besoins de levée d'anonymat ;
- les deux révocations d'anonymat sont effectuées en déchiffrant respectivement le chiffrement de s ou la valeur g^u.

Cette construction a ainsi été prouvée sûre par Hufschmitt et Traoré [HT07], dans le cas où la signature Camenisch-Lysyanskaya est celle basée sur q-SDH et en utilisant à la fois du chiffrement de Paillier [Pai99] et du chiffrement El Gamal [Gam84].

1.4.2 Signatures Partiellement Aveugles Invariables

Pour des raisons qui seront expliquées dans le chapitre suivant, notamment dans la Section 2.2 sur la gestion de l'identité, j'ai eu besoin, avec Eric Malville et Jacques Traoré [CMT08], d'introduire le concept de signature partiellement aveugle invariable. A l'instar des signatures partiellement aveugles, le signataire garde la connaissance d'une partie M du message msg qu'il est en train de signer. En supposant qu'un même utilisateur interagisse plusieurs fois avec le même signataire, nous avons par contre divisé la partie secrète du message en deux sous-parties. La première m^* est conforme à celle qui est utilisée pour les signatures partiellement aveugles. La seconde m va par contre être invariable d'une demande à une autre et va correspondre à une valeur $m = g_2^v$ où v est un secret uniquement connu de l'utilisateur[4].

Chaque utilisateur a ainsi accès à une procédure Sign pour obtenir une première signature, puis au protocole ReSign pour les suivantes prenant en entrée le même sous-message m. Nous avons alors les propriétés suivantes :

- du point de vue du modèle, la non falsification est très proche de celle décrite pour les signatures aveugles à anonymat révocable, à cela près que l'adversaire est capable d'obtenir ℓ_1 signatures à l'aide de Sign et ℓ_2 signatures à partir de ReSign : il ne doit ainsi pas être capable de fournir $\ell_1 + \ell_2 + 1$ signatures valides ;

3. Nous ne nous étions par exemple pas intéressés au mode concurrent, alors que cela est fait dans [HT07].

4. Nous travaillons ici dans un groupe cyclique $\mathbb{G}$ d'ordre connu premier ou inconnu. L'élément g et les g_i sont des générateurs aléatoires de ce groupe.

– nous avons par ailleurs repris les idées du modèle de Abe *et al.* [AF96] pour les signatures partiellement aveugles afin de définir la propriété d'aveuglement partiel. En fait, nous en avons défini deux, une portant sur la procédure SIGN et l'autre sur RESIGN. Informellement, l'adversaire choisit les parties aveugles de deux messages ainsi que deux utilisateurs. On donne ensuite à chacun des utilisateurs l'un ou l'autre des messages (ainsi, chaque utilisateur a un message différent) et l'adversaire doit être capable de retrouver l'association correcte entre les messages et les utilisateurs pour gagner l'expérience.

Après avoir formalisé ce nouveau concept, nous avons proposé deux constructions, l'une d'elle [CMT08] porte sur le protocole de Chaum et Pedersen [CP92] et l'autre [CMT09] est basé sur les signatures Camenisch-Lysyanskaya. Pour le premier, le lecteur intéressé pourra se référer à [CMT08] où se trouve la description. Pour le second, l'idée principale consiste pour l'utilisateur a produire deux engagements, l'un C' sur v et l'autre C'' sur m^* et ensuite interagir avec le signataire afin d'obtenir une signature Camenisch-Lysyanskaya sur $C = C'C''g^{\mathcal{H}(M)}$. La signature divulguée sera alors la valeur g_2^v et la preuve de connaissance de la signature Camenisch-Lysyanskaya sur v et les messages m^* et M, sans révéler v ni la signature.

1.5 Les Signatures Déléguées

Les signatures électroniques ont pour but d'assurer l'identité de l'émetteur du message mais aussi l'intégrité de ce dernier. En effet, si le message est modifié, alors la signature n'est plus valide. Dans certains cas, dont quelques un que nous verrons dans le chapitre suivant, il peut être utile qu'une entité désignée par le signataire ait le pouvoir de modifier une partie d'un message initialement signé, de telle sorte que la signature sur le message final est toujours attribuée au signataire initial.

Il existe dans la littérature deux types de modifications. La première consiste à permettre à l'entité désignée de supprimer certaines parties du message initialement signé. La seconde donne à l'entité le pouvoir de modifier certaines parties du message : ce sont les signatures "déléguées" [5]. Récemment, Brzuska *et al.* [BFF+09] ont proposé un modèle très complet sur ce type de signature. De mon côté, j'ai eu plusieurs contributions dans ce domaine, que je vais présenter dans cette section.

1.5.1 Une Nouvelle Construction

La première construction de signature déléguée est celle d'Ateniese *et al.* [ACdMT05]. Celle-ci est basée sur une brique cryptographique appelée fonction de hachage caméléon. Cette brique correspond à une fonction de hachage, basée sur des mécanismes de cryptographie à clé publique, permettant à une entité particulière en possession d'une clé secrète de trouver des collisions. Plus précisément, une valeur r est associée à chaque message m et son haché h. Elle est choisie aléatoirement lors du hachage initial et est nécessaire pour vérifier la validité du haché. Pour un message m et un haché donné h, cette valeur r est unique. La clé secrète est alors utilisée, à partir d'un haché h et d'un nouveau message $\tilde{m}$, afin de trouver la valeur $\tilde{r}$ correspondante. A partir de là, le schéma de signature déléguée [ACdMT05] est construit

5. En anglais, on parle de "sanitizable signature", ce qui est un nom pour le moins étrange sans réelle traduction française. Je choisis donc d'appeler ce nouveau type de signature les signatures déléguées.

de la façon suivante :

– le signataire va utiliser la fonction de hachage caméléon pour hacher chaque partie modifiable du message et ainsi obtenir autant de couples (h_i,r_i) qu'il y a de parties modifiable dans le message. Il remplace alors les parties modifiables du message par le haché h_i correspondant (les parties non modifiables sont inchangées) et utilise ensuite une signature classique pour signer cette nouvelle donnée notée M. La signature déléguée consiste alors en la signature standard et les r_i :

$$M = m_1\|h_2\|m_3\|h_4 \text{ et } \sigma = (\textsc{Sign}(M),\{r_2,r_4\});$$

– la modification (ou la "sanitization" en anglais) consiste, pour le délégué possédant la clé secrète de la fonction de hachage caméléon, en l'utilisation de la propriété des fonctions de hachage caméléon. Il va ainsi trouver les collisions en modifiant, le cas échéant, les parties modifiables du message et en calculant les nouvelles valeurs des $\tilde{r}_i$. De ce fait, les h_i ne sont pas modifiés, les données M sont donc identiques et la signature classique initiale est donc toujours valide :

$$M = m_1\|h_2\|m_3\|h_4 \text{ et } \sigma = (\textsc{Sign}(M),\{\tilde{r}_2,\tilde{r}_4\}).$$

Le problème avec cette construction[6] est qu'à partir de deux modifications d'un même message, il est possible de construire un nouveau message avec une signature valide, sans connaître la clé secrète du délégué. En effet, par exemple dans le cas où deux parties sont modifiables dans un message signé, et en admettant que le délégué a auparavant modifié deux fois ce message, il est possible de créer un nouveau message en associant la première modification de la première partie et la seconde modification de la seconde partie.

Avec Fabien Laguillaumie et Michel Milhau [CLM08], puis avec Amandine Jambert [CJ09], j'ai proposé une solution à ce problème. Celle-ci consiste à rajouter un élément aux données M correspondant au haché h_c du message complet calculé à l'aide de la fonction de hachage caméléon. Par exemple, nous obtenons

$$M = m_1\|h_2\|m_3\|h_4\|h_c.$$

Le délégué va devoir modifier cette valeur à l'aide de sa clé secrète puisque le message n'a plus la même valeur.

1.5.2 Extensions sur les Signatures Déléguées

Avec Amandine Jambert [CJ09], nous nous sommes aussi intéressés aux limitations que le signataire peut donner au délégué. En effet, il peut s'avérer utile, dans certains cas, d'empêcher le délégué de réaliser certaines actions, ce qui conduit à plusieurs extensions des schémas de signature délégué. Certaines de ces extensions avaient déjà été étudiées par Klonowski et Lauks [KL06]. Nous avons ainsi été les premiers à formaliser ces extensions. Nous avons pour

6. Ce n'est pas le seul mais je vais, dans ce mémoire, uniquement m'intéresser à celui-ci car il s'agit de mon apport principal. Le lecteur intéressé pourra regarder [BFF+09] pour des détails sur les autres problèmes de ce schéma.

cela étendu le modèle initial de Bruzka *et al.* [BFF+09]. Nous nous sommes plus précisément intéressés aux extensions suivantes :

- limiter les modifications possibles d'une partie modifiable du message à un ensemble de valeurs prédéfinies par le signataire. Cette extension avait été proposée dans [KL06] et nous avons prouvé que si elle était utilisée dans le cadre de notre schéma [CJ09], elle donnerait un système sûr dans notre nouveau modèle ;
- forcer le délégué à modifier de la même façon certaines parties modifiables. A nouveau, cette extension avait été étudiée dans [KL06], mais nous avons montré que leur proposition n'était pas sûre puisqu'à partir de deux modifications du même message, il est possible de récupérer la clé secrète du délégué. Nous ne sommes par contre pas parvenus à trouver une construction sûre pour cette extension ;
- limiter le nombre de modifications que le délégué est capable de faire sur les parties modifiables. Le but consiste à retrouver un secret appartenant au délégué dans le cas où ce dernier tricherait. Avec Amandine Jambert, nous avons à nouveau trouvé une faille dans la proposition de Klonowski et Lauks [KL06], puisqu'il est possible de retrouver la clé secrète du délégué après une unique modification effectuée par ce dernier. Nous avons proposé une réparation de ce système qui consiste à utiliser, comme dans [KL06], l'interpolation de Lagrange mais, contrairement à ce dernier, une fonction de hachage caméléon résistante à la divulgation de clé. Nous protégeons ainsi doublement la clé de modification puisque même si l'interpolation permet d'en révéler une partie, l'autre reste connue uniquement du délégué ;
- limiter le nombre de fois que le délégué est capable de modifier un message reçu. Nous avons proposé une solution à cette extension en utilisant des techniques utilisées dans la monnaie électronique pour retrouver l'identité d'un fraudeur ayant dépensé deux fois la même pièce. Plus précisément, la validité d'une preuve de connaissance à divulgation nulle de connaissance d'un certain secret réside en ce qu'à partir d'un engagement, de deux questions et des deux réponses associées, il est possible de retrouver le secret impliqué. Dans notre cas, les engagements sont choisis lors de la phase de signature et accumulés dans un accumulateur [CL02a, Ngu05, CKS09] qui correspond à une partie non modifiable du message signé par le signataire. La question dépend alors des modifications apportées par le délégué, ce qui donne nécessairement deux questions, donc deux réponses distinctes, et par conséquent la fuite du secret du délégué dans le cas où il utiliserait deux fois le même engagement.

1.5.3 Variantes des Signatures Déléguées

Je me suis par ailleurs intéressé aux adaptations qu'il est possible de faire aux signatures déléguées. Ces variantes sont nées de services spécifiques que je souhaitais étudier et qui seront présentés dans le prochain chapitre.

Avec Fabien Laguillaumie et Michel Milhau [CLM08], nous avons proposé un système très proche où le pouvoir de délégation peut être donné *a posteriori*. En effet, dans les constructions initiales décrites ci-dessus, le signataire doit dès le processus de signature choisir quelle clé va être utilisée par le délégué pour trouver des collisions. Dans notre système de signature déléguée à trappe, le signataire n'a pas besoin de le décider si tôt. Il va ainsi, au moment de la signature, calculer une trappe propre au message qu'il est en train de signer et qui pourra

être donnée à n'importe qui, quand le besoin s'en fera sentir. Une trappe de modification sur un message ne donne bien évidemment aucun pouvoir sur d'autres messages signés par le même signataire. La brique de base que nous avons utilisée et qui permet à notre système de fonctionner est une fonction de hachage caméléon basée sur l'identité. Le reste de la construction reste très proche du système initial avec une fonction de hachage standard.

Comme le montrent le titre et les différentes sections de mon mémoire, je m'intéresse beaucoup dans ma recherche à la protection de la vie privée et à l'anonymat des utilisateurs. Je me suis donc dans un second temps intéressé au cas où il y a plusieurs signataires et un seul délégué, et où ce dernier ne doit pas connaître l'identité du signataire initial. Ce dernier est donc anonyme vis-à-vis du délégué qui va lui-même être capable de modifier une signature obtenue. Pour cela, j'ai introduit avec Amandine Jambert [CJ10a] une brique de signature de groupe déléguée, mélangeant les deux concepts introduits précédemment. En fait, cela se passe relativement bien dans le cas où la signature de groupe est sûre dans le modèle de l'oracle aléatoire, c'est-à-dire lorsqu'elle utilise une fonction de hachage et l'heuristique de Fiat-Shamir. En effet, dans ce cas, le challenge de la signature de connaissance présente dans la signature de groupe (voir la Section 1.3) inclut le message qui va, dans notre cas, lui-même être divisé en plusieurs parties qui seront construites comme décrit précédemment.

1.6 Gestion de Clés dans les Groupes Hiérarchiques

La gestion des clés dans des groupes est une problématique relativement vieille en cryptographie. En effet, il existe de nombreux cas pratiques où tous les membres d'un groupe doivent être capables par exemple de déchiffrer un même document, sans qu'il soit nécessaire, pour des raisons évidentes, de chiffrer le document en question pour tous les membres. Il existe ainsi de nombreuses constructions permettant à plusieurs membres d'un même groupe de s'accorder sur une même clé cryptographique, la plupart d'entre elles [STW00, LKKR03, KPT04, BCP07] étant en fait une généralisation du protocole de Diffie-Hellman entre deux acteurs [DH76].

Dans certains cas pratiques, comme l'un de ceux que je présenterai dans le prochain chapitre, les membres du groupe ne sont pas tous égaux : on parle alors de groupe hiérarchique. Dans ce cas, les techniques précédentes ne fonctionnent plus et il faut donc trouver autre chose. Il existe une bibliographie relativement importante sur le sujet, avec des techniques très différentes de celles utilisées pour les groupes non hiérarchiques.

Dans [CJ08], Amandine Jambert et moi-même avons réconcilié les deux mondes en proposant un système de gestion de clés dans un groupe hiérarchique basé sur un système de gestion de clé dans un groupe non hiérarchique. Un groupe hiérarchique est traditionnellement représenté par un graphe orienté acyclique[7]. Par ailleurs, nous avons été les premiers à proposer un système permettant de gérer un graphe avec potentiellement plusieurs racines. Notre solution se résume en la manipulation de trois méthodes distinctes :

– dans le cas où un nœud ne possède qu'un seul père, nous utilisons alors un HMAC pour générer la clé du nœud fils,

$$k_{fils} = \mathrm{HMAC}(k_{pere} \| C \| c_{pere})$$

7. La représentation pourrait aussi être un arbre mais cela ne couvre pas tous les cas de figure, comme nous le verrons pour la protection des contenus dans le chapitre suivant.

où C est une constante dépendante du graphe et c_{pere} un compteur dépendant du père ;

– dans le cas où le nœud étudié a plusieurs pères, nous utilisons alors le protocole de génération d'une clé dans un groupe non hiérarchique où les membres sont les pères et la clé issue du protocole est celle du nœud fils ;

– dans le cas où plusieurs pères possèdent en commun plusieurs fils, nous avons introduit le concept de nœud virtuel (voir Figure 1.2). Nous utilisons alors les techniques ci-dessus

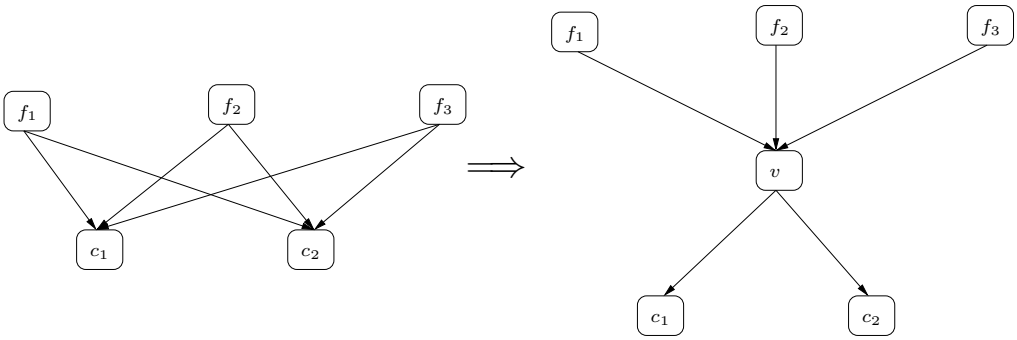

FIG. 1.2 – *Principe du nœud virtuel*

pour générer la clé du nœud virtuel et les clés des nœuds fils.

1.7 Conclusion et Perspectives

Dans ce premier chapitre, j'ai présenté les briques cryptographiques sur lesquelles j'ai apporté des évolutions intéressantes. J'ai ainsi proposé des améliorations de preuves de connaissance utilisées dans de nombreux articles existants. J'ai par ailleurs, à l'occasion de la thèse d'Iwen Coisel, proposé une preuve de sécurité d'un protocole générique de preuve de connaissance à divulgation nulle de connaissance. Celui-ci couvre une grande partie des constructions actuelles, et permet ainsi aux concepteurs de se dispenser d'en prouver la sécurité. Dans ce domaine, l'étape suivante serait d'utiliser ce type de preuve pour aider à la construction de protocoles. Il est en effet nécessaire, dans certains cas, de rajouter par exemple des engagements sur des secrets du prouveur, uniquement pour les besoins de la preuve de connaissance. En procédant étape par étape jusqu'à obtenir un schéma sûr, il est alors vraisemblablement possible de construire formellement de telles preuves de connaissance.

Une possibilité pour rendre non-interactive une preuve de connaissance est d'utiliser l'heuristique de Fiat-Shamir, et donc une fonction de hachage. De mon point de vue, cela entraîne deux inconvénients majeurs. Le premier est qu'il est nécessaire de se placer dans le modèle de l'oracle aléatoire. Le second est qu'il n'est pas possible de prouver la connaissance d'une telle signature de connaissance. Ces problématiques ont en partie été résolues par Groth et Sahai [GS08] mais ces travaux n'incluent que les environnements bilinéaires et il ne s'agit pas d'une preuve de connaissance, mais seulement d'une preuve de validité.

J'ai aussi contribué à l'élargissement des propriétés de briques cryptographiques pour l'anonymat avec de nouvelles variantes de signatures de groupe ou de signatures aveugles. Cela permet d'obtenir des innovations sur un nombre plus large de services des télécommunications,

comme nous le verrons dans le chapitre suivant. L'efficacité de ces constructions n'est pas encore optimale et je reste persuadé qu'il y a encore des améliorations possibles dans ce domaine. C'est encore plus flagrant lorsque l'on regarde les systèmes dans le modèle standard. En effet, les preuves Groth-Sahai nécessitent de nombreux éléments du groupe afin de prouver la validité d'une équation bilinéaire. Par ailleurs, comme j'ai pu le montrer dans les constructions que j'ai présentées ci-dessus, il existe de nombreuses similitudes entre les signatures Camenisch-Lysyanskaya, les signatures de groupe et les signatures aveugles. Mon avis est qu'il faut étudier plus en détail ces similitudes afin de créer des passerelles entre les unes et les autres.

Lors de la thèse d'Amandine Jambert, nous avons élargi l'utilisation des signatures déléguées en proposant plusieurs extensions ou différentes façons de les appréhender. Les signatures déléguées ne sont apparues que très récemment en cryptographie et il reste de nombreuses pistes à explorer : concevoir des schémas n'utilisant pas les fonctions de hachage caméléon, aborder d'autres extensions, trouver des constructions pour celles-ci, etc.

Chapitre 2

Protection de la Vie Privée dans les Services

Le choix de la plupart des outils cryptographiques que j'ai étudiés dans mes travaux de recherche n'est pas dû au hasard. C'était à chaque fois dans le but de proposer un service pour lequel il est nécessaire d'apporter de la sécurité et/ou la protection de la vie privée des utilisateurs. Ce chapitre va ainsi inclure mes études sur des systèmes de vote électronique [CGT06], de gestion de l'identité [CMT08], de contenus numériques [CLM08] ou de paiement anonyme [CJ10b, CJ10a]. Ces résultats sont issus de mes travaux avec mes collègues, en fonction des besoins que j'ai pu rencontrer ou des idées que j'ai pu avoir. Certains d'entre eux [CJ10b, CJ10a] sont dus à mes travaux avec Amandine Jambert pendant sa thèse sous mon encadrement.

Sommaire

Il existe de plus en plus de services dans le monde des télécommunications. Pour certains d'entre eux, comme le paiement ou simplement l'accès à des ressources, la sécurité est primordiale, que ce soit pour l'authentification, l'intégrité et/ou la confidentialité.

La protection de la vie privée est souvent vue comme une propriété supplémentaire. Elle est parfois nécessaire, comme par exemple dans le domaine de la santé où tout le monde n'a

pas accès aux données de santé des patients. C'est aussi vrai pour le vote électronique où il faut clairement qu'il ne soit pas possible de relier un électeur et son vote.

Il existe par ailleurs certains services qui risquent prochainement de tomber dans cette dernière catégorie, soit par des textes de lois spécifiques, soit par la poussée d'instances publiques (comme cela a été le cas avec la CNIL pour le système de transport NAVIGO).

Dans ce chapitre, je vais ainsi aborder des problématiques liées au vote électronique, à la protection de la vie privée dans la gestion des identités, à la protection des contenus et au paiement. La monnaie électronique, bien que rentrant dans cette dernière catégorie, ne sera abordée qu'au prochain chapitre. En effet, je possède dans ce domaine de nombreux résultats qui méritent un chapitre à part entière.

2.1 Vote Électronique

Le vote électronique a pour objectif de remplacer le vote traditionnel. Il possède de nombreux atouts par rapport à son homologue papier : le dépouillement est plus rapide, un électeur n'est pas obligé d'aller dans son bureau de vote pour s'exprimer, il est plus attractif et pourrait augmenter le taux de participation, etc.

2.1.1 Contexte Général

On distingue aujourd'hui le vote en ligne où les électeurs ont uniquement besoin d'une connexion internet pour voter, le vote hors-ligne où les électeurs doivent se déplacer dans leur bureau de vote mais font face à une machine pour voter, et le vote hybride qui permet à un électeur de se déplacer dans n'importe quel bureau de vote pour exprimer son choix, même dans le cas d'un vote local.

Dans le cas du vote hors-ligne, il y a aujourd'hui deux écoles. La première préfère les "machines à voter" qui doivent vérifier une centaine de critères bien définis pour être considérées comme sûres et utilisables dans des systèmes réels. La sécurité de ces machines ne repose en aucun cas sur des briques cryptographiques, ce qui justement est le cas pour la seconde école. Je vais ici m'intéresser à cette dernière, pour laquelle il existe, de façon très schématique, trois types de construction, en fonction de la brique cryptographique utilisée.

1. Il existe tout d'abord les constructions basées sur les réseaux de mélangeurs. Un tel système va prendre en entrée les bulletins et, à la manière du dépouillement d'un vote traditionnel sur papier, va les mélanger de telle sorte qu'il sera infaisable de faire la correspondance entre les entrées et les sorties. Il est ainsi composé de plusieurs mélangeurs qui vont, les uns après les autres, modifier les entrées et les mélanger avant de les envoyer au suivant. Pour des raisons de sécurité du système de vote, il est nécessaire que le réseau de mélangeurs utilisé soit universellement vérifiable. Chaque mélangeur doit donc prouver qu'il a correctement effectué son travail. Ceci rend le système de vote parfois peu efficace au niveau du dépouillement.
2. Il est aussi possible de construire un système de vote électronique à partir d'un schéma de chiffrement homomorphique. Dans ce cas là, les électeurs vont, indépendamment les uns des autres, chiffrer leur choix avec une clé publique unique. La propriété d'homomorphisme du schéma de chiffrement va alors permettre, au moment du dépouillement,

d'agréger l'ensemble des chiffrés de tous les électeurs en un seul chiffré qui peut alors aisément être déchiffré pour connaître le résultat final de l'élection. La difficulté d'utilisation de cette méthode est qu'elle fonctionne très bien pour un vote oui/non et qu'elle devient plus complexe lorsqu'il s'agit de voter pour un candidat parmi n.
Dans le cadre du projet SAVE (Sécurité et Audit du Vote Electronique), financé par l'ANR et auquel je participe, nous avons utilisé cette technique de façon très efficace, en utilisant le schéma de chiffrement BGN [BGN05], pour répondre aux besoins du vote préférentiel, où chaque électeur doit classer les candidats par ordre de préférence. Je ne détaille pas cette solution, ma contribution étant mineure.

3. Enfin, certains systèmes de vote électronique utilisent les signatures avec anonymat. Lors de ma thèse, j'avais introduit le concept de signature de liste, variante des signatures de groupe, pour créer un nouveau système de vote électronique [CT03a]. Je présenterai une variante de ce schéma un peu plus tard dans mon mémoire, en imaginant que l'électeur possède une carte à puce ayant des capacités restreintes.
Dans ce type de construction, la brique cryptographique la plus couramment étudiée pour réaliser un système de vote électronique est la signature aveugle. Le principe est alors le suivant : chaque électeur va dans un premier temps interagir avec une autorité habilitée pour obtenir de cette dernière une signature aveugle sur son choix. De par les propriétés des signatures aveugles, cette autorité va être incapable d'obtenir la moindre information sur le choix effectué, et ne va pas non plus être capable de reconnaître le choix ni sa signature lorsque, durant la seconde phase, l'électeur va envoyer son bulletin (c'est-à-dire son choix et la signature) à l'urne. Je vais maintenant m'intéresser plus en détails à ce type de construction.

2.1.2 Cryptanalyse de Votopia

J'ai eu l'occasion de m'intéresser au vote électronique basé sur les signatures aveugles lors de la rédaction d'un article avec Matthieu Gaud et Jacques Traoré [CGT06]. Le système de vote qui en a résulté a par la suite été utilisé lors d'un vote réel dans les Universités de Lyon et Nantes, puis lors d'un vote dans les mutuelles de santé.

Nous sommes partis du système de vote électronique Votopia [KKLA01], qui a été utilisé pour élire le meilleur joueur de la coupe du monde de football 2006 en Corée et au Japon. Celui-ci utilise un schéma de signature aveugle, un réseau de mélangeur qui ne nécessite pas d'être universellement vérifiable (voir plus haut), et un schéma de chiffrement partagé pour le dépouillement. La phase de vote est divisée en plusieurs étapes :

1. l'électeur fait son choix v et le chiffre à l'aide de l'algorithme de chiffrement afin d'obtenir $x = \textsc{Enc}(v)$. L'électeur aveugle alors ce chiffré $e = \textsc{Blind}(x,r)$ où r est une valeur aléatoire et il signature finalement e ;
2. l'autorité signataire vérifie alors la signature de l'électeur, ce qui lui permet de vérifier que c'est un électeur valide n'ayant pas encore voté, puis elle signe la valeur e afin d'obtenir d. A la fin de cette phase de vote, l'autorité va annoncer le nombre d'électeurs ayant demandé une telle signature et publie toutes les informations concernant cette phase ;
3. l'électeur est capable de désmasquer d à l'aide de r afin d'obtenir une signature y sur le chiffré x. Finalement, il prépare le couple (x,y) afin de l'envoyer au réseau de mélangeur,

obtenant ainsi l'anonymat dont il a besoin pour envoyer son bulletin à l'urne. Je note c la valeur obtenue. Celle-ci est à nouveau signée par l'électeur, puis envoyée à l'urne ;

4. l'urne publie finalement l'ensemble des bulletins obtenus.

Lors du dépouillement, chaque bulletin c (sans la signature de l'électeur) va être envoyé au réseau de mélangeurs qui va au final donner un ensemble de couples (x,y) dans un ordre complètement différent de celui donné en entrée. Chaque signature y va être vérifiée, puis chaque donnée x va être déchiffrée (si la signature correspondante y est valide) afin de retrouver le choix effectué. Le résultat peut alors être proclamé.

Nous avons montré dans [CGT06] que ce système n'est pas sûr, puisque le premier mélangeur du réseau peut tricher en obtenant un résultat différent de celui qui devrait être donné, sans que cela ne soit décelé. Il y a en fait deux cas :

1. si le nombre n de bulletins envoyés à l'urne est plus petit que le nombre N d'électeurs ayant obtenu une signature aveugle, alors il suffit au premier mélangeur de remplacer dans la liste reçue des bulletins aléatoires (a priori ceux des partisans du candidat qu'il ne souhaite pas voir élu) par des bulletins de son choix (a priori ceux du candidat de son choix) qui n'ont pas été envoyés à l'urne (en supposant qu'il a $m \leq N - n$ complices) ;
2. si $n = N$, alors le premier mélangeur peut demander à $m < N$ complices d'envoyer à l'urne des faux bulletins. Lors de la phase de dépouillement, il va ensuite remplacer m bulletins pris aléatoirement par les m réellement obtenus par ses complices lors de la phase de signature aveugle.

Dans les deux cas, le premier mélangeur du réseau est capable de modifier le résultat du vote.

2.1.3 Anonymat Révocable ou Non

Nous avons évoqué deux façons de réparer le système précédent. La façon la plus naturelle d'empêcher le premier mélangeur du réseau de frauder est de l'obliger à faire son travail correctement. Pour cela, il suffit de lui demander de prouver que les entrées qu'il obtient et les sorties qu'il donne concordent : il devient donc un mélangeur universellement vérifiable. Cependant, cela ne suffit pas. En effet, dans ce cas, il suffit aux premier et deuxième mélangeurs de former une coalition pour reproduire l'une ou l'autre des deux attaques précédentes. En effet, le premier va prouver qu'il a correctement mélangé les entrées mais il va pouvoir fournir au second la table de correspondances, ce dernier va alors frauder de la même façon que le faisait précédemment le premier mélangeur. Il faudrait alors que le deuxième mélangeur soit aussi universellement vérifiable... Et ainsi de suite, jusqu'à obtenir un réseau de mélangeurs universellement vérifiable : il n'est alors plus utile d'utiliser en plus le schéma de signature aveugle.

Nous avons trouvé une alternative où la signature aveugle a encore sa place. Plus précisément, nous avons utilisé un schéma de signature aveugle à anonymat révocable. Cela peut paraître paradoxal d'être capable de lever l'anonymat dans un système de vote où l'anonymat est primordial. Pourtant, cela entraîne simplement la perte de l'anonymat inconditionnel procuré par les schémas de signature aveugle. Mais, d'un point de vue anonymat, nous arrivons au même niveau que les solutions basées sur les réseaux de mélangeurs ou le chiffrement homomorphique. Dans ce cas, l'anonymat est conditionné par le chiffrement puisqu'il peut être levé par une coalition de mélangeurs ou par des autorités de vote.

Notre système de vote est alors très proche de celui de Votopia. En particulier, la phase de vote décrite ci-dessus est identique, à cela près que le schéma de signature aveugle est à anonymat révocable. Par contre, la phase de dépouillement est très différente. Elle fonctionne de la façon suivante :

- pour chaque électeur ayant obtenu une signature aveugle mais n'ayant pas envoyé de bulletin dans l'urne (voir la première attaque sur Votopia), nous exécutons la révocation du protocole d'obtention d'une signature aveugle (voir Section 1.4). Nous obtenons ainsi le couple (x,y) qui peut alors être mis sur liste de révocation. Ainsi, après le passage par le réseau de mélangeurs, tout couple (x,y) sur liste de révocation est annulé ;
- si jamais il reste néanmoins des couples (x,y) non valides (voir la seconde attaque sur Votopia avec les bulletins non valides des complices), et seulement dans ce cas là, nous demandons au réseau de mélangeurs de prouver la correspondance, mais uniquement sur cette entrée (il ne s'agit donc pas d'un réseau de mélangeurs universellement vérifiable), jusqu'à arriver à deux cas possibles :
  1. soit un mélangeur est déclaré coupable. Dans ce cas, la clé secrète du réseau de mélangeurs est publiée et le dépouillement s'effectue alors directement avec cette clé (voir [CGT06] pour plus de détails) ;
  2. soit le réseau de mélangeurs est honnête. Dans ce cas, on retrouve l'identité de l'électeur ayant demandé cette signature aveugle (voir Section 1.4). Le bulletin correspondant est annulé et le dépouillement peut se poursuivre.

Nous obtenons un système très intéressant en pratique. En effet, même si la phase de dépouillement semble complexe, elle nous permet d'identifier l'origine de toute fraude ou anomalie. Ainsi, il est peu vraisemblable qu'il y en ait réellement une. Il s'ensuit que le réseau de mélangeurs n'aura jamais à prouver sa bonne foi et que les révocations d'anonymat ne seront que très peu utilisées.

2.2 Application à la Gestion de l'Identité

Les internautes les plus actifs possèdent aujourd'hui de nombreuses identités sur Internet. Il peut s'avérer intéressant pour eux qu'ils aient la possibilité de simplifier leur utilisation de ces identités. Mais cela ne doit pas être fait au détriment de leur vie privée.

2.2.1 La Fédération d'Identité

La fédération d'identité permet aux internautes possédant de nombreuses identités sur internet (chacune étant utilisée pour accéder à un fournisseur de services) de les fédérer en une unique identité gérée par une entité nommée fournisseur d'identité. Ce principe, décrit par l'alliance Liberty [Lib], permet par ailleurs à un internaute d'accéder à ses différentes identités en ne s'authentifiant qu'une seule et unique fois : nous appelons cela le principe d'authentification jetable (de l'anglais "single sign-on").

Le principe est le suivant : le fournisseur d'identité va référencer chaque identité d'un même utilisateur avec un alias. Ce dernier va donc faire le lien entre l'identité de l'utilisateur chez le fournisseur d'identité et l'identité de ce même utilisateur chez le fournisseur de services. La Figure 2.1 donne un exemple d'alias entre un fournisseur d'identité et deux fournisseurs

de service. L'étape de fédération d'identité permet de créer cet alias. Chaque fournisseur (de

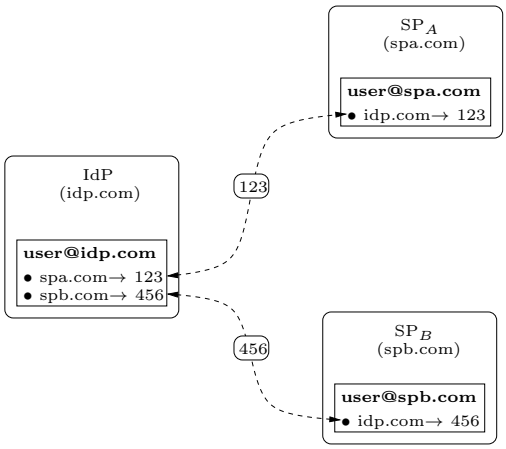

FIG. 2.1 – *Principe de la fédération d'identité*

services et d'identité) va authentifier de son côté l'utilisateur et le fournisseur d'identité va au final signer une requête d'authentification stipulant le nouvel alias, des données propres à la requête et le fait qu'il a effectivement authentifié l'utilisateur. Le fournisseur de services et le fournisseur d'identité vont alors stocker dans leurs bases respectives cet alias.

Un fournisseur de services souhaitant par la suite identifier et authentifier un utilisateur va lancer une requête d'authentification incluant sa propre identité et l'alias de fédération. Il va ensuite signer cette requête. Celle-ci va ensuite être envoyée au fournisseur d'identité qui va retrouver l'identité de l'utilisateur (à partir de l'alias et de l'identité du fournisseur de services), va authentifier ce dernier (à l'aide d'un mot de passe ou d'un moyen d'authentification plus fort) et va signer la réponse d'authentification incluant à nouveau l'alias, des données propres à la requête et le fait qu'il a effectivement authentifié l'utilisateur.

Ce principe de fédération d'identité permet d'avoir un premier niveau de protection de la vie privée puisqu'un même utilisateur est connu avec des identités différentes auprès de chaque fournisseur de services. Ainsi, ces derniers sont incapables de recouper les informations qu'ils connaissent sur un même utilisateur. Par contre, le problème vient du fait que le fournisseur d'identité connaît toute l'information pour corréler les identités d'un même utilisateur.

2.2.2 Amélioration de la Protection de la Vie Privée

J'ai proposé avec Eric Malville et Jacques Traoré [CMT08, CMT09] un système où le fournisseur d'identité n'est plus capable de corréler les différentes identités d'un même utilisateur. Pour parvenir à cela, nous avons fait en sorte que le fournisseur d'identité ne connaisse pas les alias des utilisateurs. Pour autant, il doit les signer pour prouver qu'il a effectivement vérifié l'identité de l'utilisateur. La signature aveugle semble donc toute désignée pour répondre à ce besoin. Nous obtenons donc le principe de fédération d'identité présenté dans la Figure 2.2.

Pourtant, n'importe quel schéma de signature aveugle ne convient pas forcément. En effet, le fournisseur d'identité doit signer :

– des données permettant de tracer l'utilisateur, telles que la date, l'identifiant de requête,

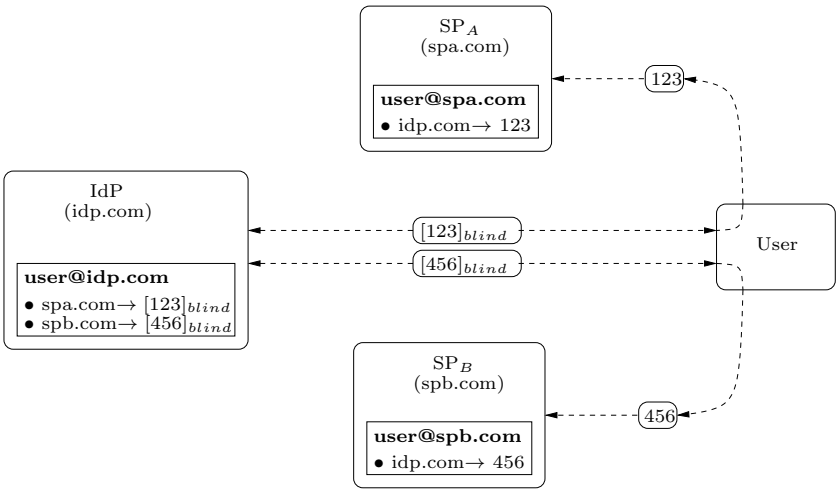

FIG. 2.2 – *Principe de la fédération d'identité "aveugle"*

etc. Ces données ne doivent pas être connues du fournisseur d'identité et seront donc masquées ;

– des données générales telles que les balises SAML [CKPM05] ou l'identité du fournisseur de services. Ces données ne sont pas sensibles et certaines doivent nécessairement (comme l'identité du fournisseur de services) être connues par le fournisseur d'identité. Nous avons donc besoin d'un schéma de signature partiellement aveugle ;

– l'alias de fédération qui ne doit bien évidemment pas être connu du fournisseur d'identité. Cette donnée a par ailleurs une propriété très particulière puisqu'elle ne doit pas varier d'un protocole (dans notre cas une session) à l'autre (entre une fédération d'identité et une authentification jetable ou entre deux authentifications jetables). Il nous faut donc utiliser un schéma de signature partiellement aveugle invariable, tel que décrit dans la Section 1.4.

Je donne l'exemple du protocole d'authentification jetable dans la Figure 2.3.

Pour réaliser en pratique un tel système, l'utilisateur doit être en mesure de retrouver les éléments concernant l'alias (voir en particulier le protocole de signature partiellement aveugle invariable dans la Section 1.4). Or, la fédération d'identité est initialement faite pour que l'utilisateur n'ait rien à stocker en local. Il est alors possible soit d'imaginer un client un peu plus lourd (avec des capacités de stockage et de calcul) comme l'alliance Liberty est en train de concevoir, soit d'utiliser par exemple un générateur pseudo-aléatoire tel un HMAC prenant en entrée les identités des fournisseurs de services et d'identité et un mot de passe potentiellement court, mémorisable par l'utilisateur.

2.3 Contenus et Protection de la Vie Privée

Les contenus numériques sont aujourd'hui omniprésents dans nos vies. L'accès à ces contenus doit bien souvent être protégé pour que seules les entités autorisées soient en mesure de les lire. Dans ce contexte, mes recherches m'ont conduit à étudier l'accès à des contenus au

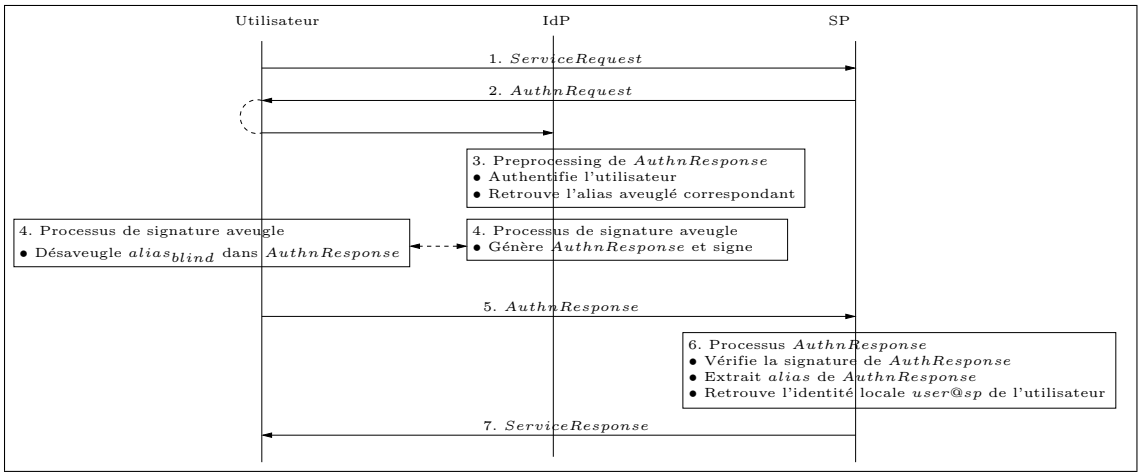

FIG. 2.3 – *Authentification jetable*

sein de groupes tels que la famille, les associations, les universités ou les entreprises.

2.3.1 DRM pour les Groupes

Les DRM (Digital Right Management) ont jusqu'à il y a très récemment été utilisées dans de nombreux systèmes de contenus numériques, afin d'en protéger l'accès. Le principe est qu'une entité émettrice d'un contenu va chiffrer ce dernier à l'aide d'une clé qui sera commune à toutes les personnes désirant lire ce contenu. Lorsqu'une personne va vouloir accéder à ce contenu, l'entité émettrice va alors créer une licence propre à ce contenu et à cet utilisateur. Celle-ci est constituée des éléments suivants :

- un numéro de licence ;
- l'identifiant du contenu ;
- les droits que l'utilisateur a sur ce contenu (lecture, gravure, copie, etc.) ;
- la clé chiffrant le contenu, elle-même chiffrée avec la clé publique de l'utilisateur ;
- la signature de l'entité émettrice sur l'ensemble des champs précédents.

Cette licence est donc propre à l'utilisateur qui en fait la demande, sans possibilité de transfert.

Dans le cas d'un groupe de personnes, par exemple la famille, il se peut que le chef de famille achète des contenus pour l'ensemble de la famille, pour seulement une partie de ces membres ou pour lui seul. En considérant qu'il est le seul interlocuteur de l'entité émettrice des contenus, il doit donc pouvoir transférer sa capacité à lire un contenu aux autres membres du groupe. Une solution simple consiste pour le chef de famille à donner sa clé de déchiffrement à tout le monde, mais cela peut poser problème dans certains cas. Une autre solution serait d'utiliser du surchiffrement de la licence par le chef de famille.

Avec Fabien Laguillaumie et Michel Milhau [CLM08], nous nous sommes donnés un contrainte supplémentaire relativement naturelle d'un point de vue implémentation, à savoir que le lecteur de contenus ne doit pas voir la différence entre un contenu acheté pour le groupe et un contenu acheté pour une personne seule. Ainsi, dans le cas de la famille, le fils

de la famille doit pouvoir être capable d'acheter seul un contenu protégé, sans avoir à passer par le chef de famille.

Notre solution consiste alors à utiliser les signatures déléguées. L'entité émettrice va produire une signature déléguée de la licence ci-dessus, où la partie portant sur la clé chiffrant le contenu est une partie modifiable. Cela peut aussi être le cas pour les droits, mais dans ce cas avec des restrictions obligeant le délégué à modifier cette partie du message selon un ensemble de messages prédéfinis. Le rôle du délégué est ici joué par le chef de famille qui va donc être capable, au moyen d'une clé secrète qui lui est propre (soit la sienne, soit relative à ce contenu avec un schéma à trappe, voir Section 1.5), de modifier les parties modifiables ci-dessus pour chiffrer la clé chiffrant le contenu pour le membre du groupe qui va pouvoir accéder à ce contenu. Nous avons ainsi rempli toutes les propriétés désirées.

2.3.2 Gestion des Contenus dans les Groupes

Imaginons maintenant qu'un groupe d'entités ait à sa disposition un certain nombre de contenus. Si nous prenons l'exemple de l'entreprise, il y a nécessairement des contenus accessibles par certaines entités et pas par d'autres, et inversement. En effet, un chef d'unité va avoir accès à des ressources qui ne sont pas nécessairement disponibles pour les membres de son unité. Par ailleurs, dans le cadre de certains projets, notamment ceux mettant en œuvre des partenariats externes, certains membres de l'équipe vont avoir accès à des documents confidentiels provenant d'autres partenaires, ce qui ne sera pas le cas du chef d'unité. De ce fait, une telle hiérarchie va être représentée par un graphe orienté sans cycle, et a priori avec plusieurs racines.

Avec Pascal Avondes, Amandine Jambert et Michel Milhau [ACJM10], nous avons ainsi proposé une architecture complète de gestion des contenus dans une groupe. Celle-ci utilise le schéma de gestion de clés dans un groupe hiérarchique décrit dans le chapitre précédent, et qui s'adapte très bien à ce cahier des charges.

2.4 Paiement Anonyme

Le paiement électronique est désormais omniprésent dans nos vies numériques. Toujours dans le cadre de la protection de la vie privée, il peut parfois être intéressant de faire en sorte qu'un internaute ne soit pas traçable dans ses paiements. Dans le chapitre suivant, j'aborderai la problématique de la monnaie électronique mais je voudrais ici dire quelques mots sur des travaux récents que j'ai menés avec Amandine Jambert. Je vais ainsi aborder les problématiques de facturation anonyme et d'abonnement anonyme à plusieurs services.

J'ai par ailleurs eu l'occasion de m'intéresser aux cartes de fidélité, avec Fabrice Clerc et Benjamin Morin [CCM06]. Le système étudié avait pour but de proposer une solution permettant aux utilisateurs d'utiliser leur téléphone mobile pour stocker et gérer leurs différentes cartes de fidélités. Nous avions par ailleurs pris en considération la possibilité pour les marchands de faire des partenariats à l'aide de cette carte universelle.

2.4.1 Facturation Anonyme

Imaginons qu'une entité propose plusieurs services différents à ses clients. Elle possède une grille de facturation qui décrit, pour chaque service qu'elle propose, le montant qu'il faut payer pour y accéder [1]. Les utilisateurs peuvent ici s'enregistrer pour être capables d'accéder aux services proposés et ils seront facturés après avoir "consommé" le service en question.

Nous partons maintenant du principe qu'un client ne souhaite pas être tracé par le fournisseur de services : il ne veut donc pas que ce dernier sache quels services l'intéressent. Nous introduisons dans un premier temps un service de facturation qui aura pour rôle d'envoyer les factures aux clients du fournisseur de services. Pour les mêmes raisons, le client ne souhaite pas que cette nouvelle entité fasse la corrélation entre les services consommés. Le fournisseur de services a donc la connaissance du service, mais pas du consommateur, alors que le fournisseur de facturation connaît le consommateur (pour pouvoir lui envoyer la facture), mais pas le service consommé. Bien évidemment, le client ne doit pas, après coup, être capable de contester la consommation du service en question.

Par ailleurs, pour que le système fonctionne, il est nécessaire que le fournisseur de services fasse parvenir au fournisseur de facturation l'information qu'un client, qu'il ne connaît pas, vient de consommé un service qu'il n'a pas le droit de révéler. Pour répondre à ce besoin un peu paradoxal, nous avons utilisé avec Amandine Jambert [CJ10a] notre brique de signature de groupe déléguée :

- le client va produire une signature de groupe déléguée où parmi les parties modifiables, il va intégrer la référence du service demandé. Les autres parties non modifiables vont par exemple inclure l'identité du fournisseur de services ou les balises `http`.Comme il s'agit d'une signature de groupe, le client va être anonyme vis-à-vis du fournisseur de services qui va seulement apprendre qu'il s'agit d'un client valide connu par le service de facturation ;
- le fournisseur de services va jouer le rôle du délégué et va donc être capable de modifier, dans le message signé, la référence du service demandé par le code de facturation correspondant (à l'aide de sa grille de facturation) ;
- de par la propriété des signatures de groupe déléguées, la signature portant sur le nouveau message contenant le code de facturation est toujours attribuée au client initialement signataire. Ainsi, le fournisseur de facturation, en tant que manageur de révocation, va être capable de lever l'anonymat de la signature afin de retrouver l'identité du client.

2.4.2 Abonnement Anonyme

Nous nous plaçons à nouveau dans le cas où une entité propose plusieurs services distincts à ces clients. En revanche, contrairement au schéma précédent, il est nécessaire de s'abonner pour être capable d'utiliser un service. A nouveau, le client va souhaiter être intraçable vis-à-vis du fournisseur de services. Nous nous retrouvons dans le cas où, lors d'une première phase, un client va interagir avec le fournisseur de services afin de s'abonner à un ou plusieurs services. Il va par ailleurs pouvoir, lors d'une éventuelle seconde phase, rajouter ou enlever des services à son abonnement, à nouveau en interagissant avec le fournisseur de services.

1. Pour que notre étude ait un intérêt, il est important, pour des raisons relativement évidentes, que plusieurs services soient facturés de la même façon.

Enfin, il va de nouveau interagir avec ce dernier lorsqu'il va accéder à un service souscris, afin de lui prouver qu'il a bien les droits pour le faire.

Le client est nécessairement anonyme et intraçable lors de l'utilisation d'un service. Il doit cependant prouver qu'il a les droits pour le faire. Avec Amandine Jambert [CJ10b], nous avons alors utilisé les signatures Camenisch-Lysyanskaya pour construire notre système (voir Section 1.2). Lors de la phase d'abonnement, le client va obtenir du fournisseur de services une signature Camenisch-Lysyanskaya sur les services auxquels il souscrit, ainsi que sur une donnée secrète qui lui est propre (voir ci-dessous). Avant d'accéder à un service, le client va prouver qu'il a les droits pour le faire en produisant une preuve de connaissance à divulgation nulle de connaissance qu'il connaît une signature sur un certain nombre de services, dont l'un deux, qui est révélé, correspond à celui qui est demandé. Si le client souhaite par la suite rajouter des services à son abonnement, il va alors utiliser le protocole permettant à un utilisateur ayant préalablement obtenu une signature Camenisch-Lysyanskaya auprès du signataire, d'interagir à nouveau avec lui pour rajouter des blocs de message. Nous avons ainsi toutes les propriétés demandées.

Nous avons par ailleurs étudié diverses possibilités de protection de la vie privée de ce client :

1. le fournisseur de services sait faire le lien entre les différentes demandes d'abonnement. Il peut alors conserver dans sa base les services précédemment souscris (pour lesquels le client peut lui prouver qu'il possède bien une signature sur ces derniers) et produire une nouvelle signature Camenisch-Lysyanskaya sur les anciens et les nouveaux services ;
2. le fournisseur de services ne sait pas faire le lien entre les différentes demandes d'abonnement. Il suffit alors pour le client de cacher les blocs de message de la signature précédemment obtenue ;
3. le fournisseur ne connaît qu'un service souscrit à la fois. Il est alors nécessaire de répéter le protocole de mise à jour d'une signature Camenisch-Lysyanskaya autant de fois qu'il y a de services nouvellement souscrits.

Nous avons enfin étudié le cas où le client est anonyme vis-à-vis du fournisseur de services au moment de la souscription. Il faut alors utiliser une signature avec anonymat, comme par exemple une signature de groupe, et utiliser le secret `usk` du schéma de signature de groupe (voir Section 1.3) comme donnée secrète. Celle-ci sera donc signée par le fournisseur de services lors de la phase de souscription. Au moment de la consommation d'un service, le client devra par ailleurs prouver que la clé `usk` présente dans la signature de groupe est la même que celle qui a été signée par le fournisseur de services. Je note ici que compte tenu de cette remarque, il est possible d'utiliser un schéma de signature de groupe déléguée, et donc le système présenté ci-dessus pour la phase de souscription (celle-ci sera facturée au client).

2.5 Conclusion et Perspectives

J'ai été confronté à de nombreux services par le biais de mon statut d'ingénieur de recherche dans une entreprise de télécommunications et il y en aura certainement d'autres. Internet continue d'évoluer et de nouveaux services sur téléphone mobile vont certainement apparaître dans les prochaines années. A cela s'ajoute la problématique de la protection de la vie privée qui prend elle aussi de plus en plus d'importance. Les directives dans ce domaine

deviennent de plus en plus pressantes et les fournisseurs de services se doivent de chercher des solutions. L'Union européenne ou l'Etat français financent de plus en plus de projets autour de ces problématiques, prouvant que le domaine est aujourd'hui porteur et qu'il reste beaucoup de perspectives.

Pour le vote électronique, j'ai essayé de proposer des systèmes pratiques répondant aux contraintes légales. Je ne pense pas que nous voterons très prochainement par internet pour du vote institutionnel. Mais pour certains votes moins sensibles (vote étudiant, dans les associations, dans les entreprises, etc.), cela peut être une bonne alternative : la recherche doit continuer dans ce domaine. La vérification universelle qu'une élection s'est correctement déroulée ou bien la certitude que le choix que l'on fait sur un écran est bien celui est réellement compilé par l'entité qui effectue les calculs cryptographiques sont deux problématiques primordiales pour que le vote électronique se généralise enfin.

Comme nous avons pu le voir, l'anonymat et l'intraçabilité de l'utilisateur lorsqu'il accède à des services web est aujourd'hui une mine de problématiques. Nous avons cherché à en résoudre certaines avec Amandine Jambert, notamment lorsque cet accès est payant, mais il y a encore des améliorations à apporter. Par ailleurs, la protection de la vie privée dans les systèmes de gestion de l'identité peut encore aller plus loin. On pourrait ainsi imaginer l'anonymat du fournisseur de services (respectivement du fournisseur d'identité) par rapport au fournisseur d'identité (respectivement au fournisseur de services). Ceci donnerait une très grande protection de la vie privée aux clients des fournisseurs de services. Mais est-ce que ces derniers accepteront ces systèmes, si ceux-ci ne leur permettent plus de faire du profiling sur les usages de leurs clients ?

La protection de la vie privée peut aussi être étudiée dans le cadre de services de proximité. Un utilisateur va par exemple obtenir des ristournes en fonction de ses attributs : lieu de résidence, âge, handicapé ou non, étudiant ou non, etc. Il faut alors étudier les briques cryptographiques permettant d'assurer l'exactitude des informations demandées, tout en protégeant la vie privée des utilisateurs, c'est-à-dire en ne révélant que le strict minimum au fournisseur de services. Je pense qu'il s'agit d'un domaine porteur et c'est pourquoi je propose cette année un nouveau sujet de thèse sur les attributs anonymes, dans le but d'étudier ces briques et de proposer de nouvelles constructions complètes.

J'ai enfin abordé dans ce chapitre la problématique de protection des contenus. Les DRM ont aujourd'hui tendance à être moins utilisées, mais cela ne veut pas dire qu'il n'est pas nécessaire de protéger l'accès aux contenus, ne serait-ce que dans le cas des contenus autoproduits. Les techniques que j'ai mises en place donnent un premier niveau de sécurisation. Il faudrait maintenant permettre à de tels systèmes de répondre à de nouveaux désirs des utilisateurs, tels que l'accès aux contenus chez des amis, ou depuis n'importe quel lecteur autorisé, où que l'on soit.

Chapitre 3

Monnaie Électronique

Cette section présente les travaux que j'ai réalisés dans le domaine de la monnaie électronique. Ceux-ci ont commencé à la fin de ma thèse, avec un article à la conférence ACISP avec Jacques Traoré. J'ai par la suite continué mes travaux dans ce domaine avec Aline Gouget, d'abord lors de son post-doc à Orange Labs, puis par le biais de projets collaboratifs (eCrypt, Crypto++ et PACE). Cette collaboration a jusqu'à présent donné des résultats majeurs dans ce domaine avec 7 publications sur la dépense efficace [CGH06, CGH07, CG07, CDG+09, CG10] et sur la monnaie transférable [CGT08, CG08]. J'ai aussi contribué avec Emeline Hufschmitt, lors de sa thèse à Orange Labs, à deux articles [CGH06, CGH07]. Par ailleurs, le montage et la gestion du projet collaboratif PACE sont liés à ces travaux et, alors que le projet n'est démarré que depuis 1 an, les réunions de travail avec les partenaires du projet ont déjà permis de rédiger un article commun [CDG+09].

Sommaire

La monnaie électronique a pour but d'émuler la monnaie traditionnelle que chacun utilise quotidiennement, tout en conservant les propriétés de sécurité du système traditionnel, comme par exemple l'impossibilité de créer de la fausse monnaie. En outre, comme nous avons affaire à une donnée numérique, donc aisément duplicable, il est nécessaire de se prémunir contre l'attaque simple qui consisterait à copier plusieurs fois une pièce de monnaie afin de la dépenser plusieurs fois auprès de marchands (ou d'utilisateurs différents) : c'est ce qui est

communément appelé la "double dépense". Une conséquence directe est qu'il doit être impossible d'accuser faussement quelqu'un d'honnête d'avoir fait une telle double-dépense. Pour finir, les deux dernières propriétés de sécurité qu'un système de monnaie électronique doit vérifier sont, d'une part, l'anonymat des utilisateurs et, d'autre part, l'impossibilité de relier deux dépenses d'un même utilisateur entre elles. En effet, lorsqu'une personne va acheter son journal chez un marchand de journaux en utilisant de la monnaie fiduciaire, ce dernier ne sait pas (nécessairement) à qui il a affaire. Et si cette même personne va ensuite chez le boulanger en utilisant un autre billet retiré auprès de la même banque, cette dernière n'a aucun moyen de relier les deux dépenses, même si elle sait quel compte vient d'être débité.

Le concept de monnaie électronique a été introduit par Chaum dans les années 1980. Depuis, la recherche en cryptographie s'y est largement intéressée et de nombreux systèmes ont vu le jour dans les plus grandes conférences, mais sans pour autant qu'un marché existe réellement aujourd'hui. De mon point de vue, l'intérêt d'une recherche active sur ce sujet est triple. Tout d'abord, l'avènement des téléphones mobiles comme "boîte à tout faire" et l'apparition du NFC (*Near Field Communication*) vont susciter un nouveau besoin. En parallèle, la protection de la vie privée devient essentielle pour la plupart des nouvelles applications des télécommunications et les systèmes vont progressivement devoir se renforcer pour être conformes aux contraintes légales, comme c'est le cas dans le domaine de la santé. Des systèmes commencent à être développés et utilisés, notamment dans les pays asiatiques toujours friands de nouvelles technologies. Ensuite, le concept cryptographique de monnaie électronique, c'est-à-dire la possibilité de dépenser anonymement et de façon intraçable un élément précédemment validé par une entité habilitée, se transpose dans d'autres services similaires tels que le "ticketing" ou les coupons. Dans les deux cas, le principe consiste à remplacer les tickets ou les coupons que l'on peut utiliser dans la vie de tous les jours par des données numériques. J'ai ainsi proposé dans [CGH06], avec Aline Gouget et Emeline Hufschmitt, un système de coupon utilisant les mêmes techniques que notre système de monnaie électronique [CGH07]. Enfin, et c'est sans doute le point le plus important pour la recherche cryptographique aujourd'hui, les systèmes créés par les cryptologues font la plupart du temps appel à des briques de base qui ont un intérêt dans d'autres domaines que la monnaie. Le fait d'améliorer les systèmes de monnaie électronique permet soit de créer de nouvelles briques, soit de porter un intérêt nouveau à des briques existantes, soit d'en améliorer certaines, faisant ainsi avancer la recherche dans ce domaine, indépendamment de la monnaie électronique. Ce constat est d'ailleurs le point de départ du projet collaboratif PACE que j'ai monté et dont j'assure la responsabilité.

3.1 A la Recherche de l'Efficacité

Un système de monnaie électronique se doit de décrire une procédure de retrait (resp. de dépense) de pièce(s) de monnaie. Pendant de nombreuses années, les cryptologues n'ont pas réussi à faire mieux que de décrire des systèmes permettant de retirer (resp. dépenser) les pièces une à une, sans lien réel entre elles. Il a fallu attendre le milieu des années 2000 et le système "*compact e-cash*", proposé par Camenisch, Hohenberger et Lysyanskaya [CHL05], pour qu'un système propose aux utilisateurs de retirer plusieurs pièces de monnaie électronique aussi efficacement que s'ils en retiraient une seule.

3.1.1 Principe des Systèmes Actuels

La difficulté d'atteindre une telle propriété vient principalement de la propriété d'intraçabilité des dépenses. En effet, il ne doit pas être possible de savoir si deux pièces de monnaie dépensées proviennent du même retrait ou non. Pourtant, si ces deux pièces ont été retirées au même moment, elles proviennent nécessairement d'un unique élément initial qui a été validé par la banque.

Ainsi, lors de ma thèse, j'avais, avec Jacques Traoré [CT03b], introduit l'idée d'obtenir, lors du retrait, une signature σ de la banque sur un élément secret s, nommé numéro de série, connu uniquement de l'utilisateur. Lors de la dépense, l'utilisateur devait alors faire une preuve de connaissance à divulgation nulle de connaissance d'une signature de la banque sur cet élément secret, sans révéler ni l'élément secret, ni la signature. Nous utilisions alors des techniques connues pour les signatures de groupe.

De ce point de vue, le système *compact e-cash*, dû à Camenisch, Hoenberger et Lysyanskaya [CHL05], a utilisé cette idée. En effet, lors du retrait, l'utilisateur va obtenir, auprès de la banque, une signature σ de type "Camenisch-Lysyanskaya" de trois éléments, u, s et t :

- le premier, u, est la clé secrète de l'utilisateur, liée à une clé publique $\mathsf{upk} = g_0^u$ connue de tous ;
- le second, s, joue sensiblement le même rôle que celui de mon système ci-dessus ;
- le troisième, t, sera utilisé en cas de double dépense, pour retrouver le fraudeur.

Cette signature est obtenue lors d'un protocole interactif entre un utilisateur et un marchand, de telle sorte que la banque n'apprend rien sur u, s et t, mais participe à la création de s.

Le principe de *compact e-cash* consiste maintenant à diversifier les secrets s et t de telle sorte qu'ils puissent être utilisés plusieurs fois, en l'occurrence 2^ℓ, de façon identique, et de telle sorte qu'il est infaisable de faire le lien entre deux diversifications différentes. Pour cela, il faut utiliser une fonction de diversification utilisant en entrée l'élément s (resp. t) et un élément j qui va varier entre 0 et $2^\ell - 1$. Cette fonction doit par ailleurs avoir les propriétés suivantes : (i) il est infaisable de faire un lien entre la sortie de cette fonction pour le même s (resp. t) et deux j différents et (ii) cette fonction est vérifiable. Il s'agit donc d'une fonction pseudo-aléatoire vérifiable (VRF en anglais). Dans *compact e-cash*, les auteurs vont utiliser celle proposée par Dodis et Yampolskiy dans [DY05].

La diversification de s va donner le numéro de série de la pièce dépensée $S = g^{\frac{1}{s+j+1}}$ où g est un générateur d'un groupe cyclique $\mathbb{G}$ d'ordre premier et j est le compteur inclus dans $[0,2^\ell[$, qui correspond au numéro de la pièce dépensée parmi l'ensemble des pièces retirées. Ce numéro de série sera utilisé par la banque pour détecter une éventuelle double-dépense. Le secret t est utilisé pour générer le tag de sécurité qui servira à retrouver l'identité du double-payeur le cas échéant. Ce tag de sécurité est ainsi calculé par $T = \mathsf{upk}\big(g^{\frac{1}{t+j+1}}\big)^R$ où $\mathsf{upk} = g_0^u$ est la clé publique de l'utilisateur et $R = \mathcal{H}(\text{data}\|\mathsf{mpk})$ est une valeur aléatoire calculée à partir d'éléments publics.

La dépense d'une pièce correspond donc à l'envoi du numéro de série S, du tag de sécurité T et de la signature de connaissance U qui s'écrit comme suit :

$$\textsc{Pok}(u,s,t,j : S = g^{\frac{1}{s+j+1}} \wedge T = g_0^u\big(g^{\frac{1}{t+j+1}}\big)^R \wedge \sigma = \textsc{Sign}(u,s,t) \wedge j \in [0,2^\ell[).$$

La preuve de connaissance de l'élément j tel que $j \in [0,2^\ell[$ est nécessairement une preuve exacte. Comme nous avons pu le voir précédemment, il existe plusieurs méthodes pour y parvenir. Celle préconisée par les auteurs de *compact e-cash* est celle proposée par Fabrice Boudot. Nous verrons dans la section suivante qu'il est également possible d'utiliser d'autres méthodes.

3.1.2 Le Cas de la Dépense Efficace

En proposant une nouvelle façon d'aborder la monnaie électronique, le schéma *compact e-cash* présenté ci-dessus a été précurseur de nombreux autres articles dans le domaine. Mais il n'aborde bien évidemment pas tous les aspects de la monnaie électronique.

Le premier reproche que nous pourrions faire est qu'il n'est pas possible de choisir le nombre de pièces que l'on souhaite retirer, puisque celui-ci est forcément égal à 2^ℓ. De plus, la valeur monétaire de la pièce retirée est nécessairement unique et ne peut pas être choisie par l'utilisateur. Dans [CGH06], Aline Gouget, Emeline Hufschmitt et moi-même avons proposé un schéma qui permet de pallier ces deux inconvénients, basé sur le système initial.

Le fait de permettre à l'utilisateur de choisir le nombre de pièces qu'il souhaite retirer n'est pas aussi simple qu'il y paraît. En effet, cela ne doit pas compromettre l'anonymat lors de la dépense d'une pièce de ce porte-monnaie. En particulier, il ne faut pas que la connaissance que la banque possède sur le nombre de pièces retirées puisse lui permettre d'apprendre quoi que ce soit sur la future dépense. Pour autant, il ne faut pas non plus que l'utilisateur (et même plusieurs utilisateurs formant une coalition) puisse dépenser plus que pièces qu'il n'en a retiré. La façon dont nous avons procédé dans [CGH06] consiste à obtenir de la banque, lors du retrait, la signature sur le nombre de pièces retirées J, en plus des autres éléments. La signature de type Camenisch-Lysyanskaya porte donc dorénavant sur la clé secrète utilisateur u, les éléments s et t et enfin sur J. A noter que cette dernière valeur est nécessairement connue (et même injectée dans le message à signer) par la banque signataire.

Par contre, pour les besoins en anonymat, il convient, lors de la dépense, de ne pas révéler cette valeur J. Dans le cas contraire, il serait possible de faire des groupes de personnes ayant retiré le même nombre de pièces, et donc de perdre en anonymat. La preuve de connaissance U du système *compact e-cash* doit donc maintenant inclure une preuve que la pièce j dépensée appartient à $[0,J[$ où j et J ne doivent pas être révélées. Nous utilisons alors notre preuve de connaissance présentée dans la Section 1.1.1 qui utilise la décomposition binaire du secret.

$$\textsc{Pok}(u,s,t,j,J : S = g^{\frac{1}{s+j+1}} \wedge T = g_0^u \left(g^{\frac{1}{t+j+1}}\right)^R \wedge \sigma = \textsc{Sign}(u,s,t,J) \wedge j \in [0,J-1[).$$

Il est maintenant possible de modifier ce système pour inclure des pièces de valeurs différentes. Lors du retrait, l'utilisateur va choisir le nombre de pièces qu'il souhaite retirer, pour chaque valeur de pièce possible (par exemple 3 pièces de 10 euro et 2 pièces de 20 euro). Pour cela, nous avons utilisé la technique décrite ci-dessus, en permettant à l'utilisateur d'obtenir une signature sur les éléments habituels u, s et t, mais aussi sur les valeurs $J_1,\ldots,J_t$ si l'utilisateur retire J_1 pièces de valeur v_1, ..., et J_t pièces de valeur v_t.

Lors de la dépense d'une pièce, l'utilisateur va procéder comme précédemment, donnant la valeur de la pièce qu'il est en train de dépenser et cachant à la fois le rang de cette pièce et le nombre de pièces de cette valeur qu'il a retirées. Pour autant, si l'utilisateur dépense sa

première pièce de valeur v_1, puis sa première pièce de valeur v_2, il va a priori calculer deux fois le numéro de série $S = g^{\frac{1}{s+1+1}}$. Notre idée consiste alors à utiliser un générateur g différent pour chaque valeur monétaire, ceci permettant de ne pas avoir à augmenter l'intervalle dans lequel doit être le compteur j.

Un tel système est une réponse partielle à la problématique de dépense efficace. En effet, si un utilisateur possède le nombre de pièces qu'il désire, selon plusieurs valeurs monétaires, il est alors capable de faire l'appoint pour dépenser exactement la somme demandée. La dépense sera ainsi optimisée puisque l'utilisateur se servira de la pièce de valeur la plus proche de ce qui lui est demandé. Pourtant, si on demande 30 euro à l'utilisateur alors qu'il ne possède que des pièces de 10 et 20 euro, il devra alors faire deux dépenses indépendantes. Nous n'avons donc pas tout résolu.

Le fait qu'il n'y ait qu'un seul porte-monnaie contenant plusieurs pièces, et donc une représentation compacte de plusieurs pièces en même temps, laisse à penser qu'il est possible de simplifier la dépense de plusieurs pièces provenant du même porte-monnaie. Une première étape a été effectuée dans [ASM07] puisque les auteurs montrent, entre autres, qu'il est possible de mutualiser les preuves de connaissance de la signature et de l'appartenance de j à l'intervalle $[0,2^\ell[$ (qui est l'opération la plus coûteuse). Une solution plus intéressante, trouvée par Cécile Delerablée dans le cadre du projet Crypto++ auquel j'ai participé, permet en plus, par rapport à la preuve précédente, de mutualiser la preuve sur les numéros de série. Pour cela, l'idée est d'utiliser de la vérification de type batch et de se servir du lien qu'il existe entre un numéro de série et le suivant ($S = g^{\frac{1}{s+j+1}}$ et $S = g^{\frac{1}{s+(j+1)+1}}$). Pour autant, cette technique ne dispense pas l'utilisateur de calculer tous les numéros de série. Par ailleurs, l'étude a été effectuée dans le cadre de la monnaie électronique "équitable", qui nécessite qu'une entité habilitée soit capable, à tout moment, de lever l'anonymat d'une dépense. Elle n'inclut donc pas l'optimisation pour le tag de sécurité, qui paraît plus compliquée.

Finalement, les articles existants sur le sujet ne sont pas parvenus à obtenir une dépense qui ne soit pas linéaire en le nombre de pièces dépensées, sauf si on dépense tout le porte-monnaie en une seule fois [ASM07]. La difficulté ne vient cette fois-ci pas du tag de sécurité mais, comme expliqué plus haut, de la nécessité apparente de calculer et d'envoyer au marchand tous les numéros de série des pièces dépensées. Les travaux publiés portent sur la simplification de la preuve de validité des pièces dépensées mais rien n'a (encore) été trouvé sur les numéros de série.

3.1.3 Une Nouvelle Vision

Dans le cadre du projet PACE, au vu de l'échec relatif des précédents résultats, nous avons décidé de revoir la façon d'appréhender la monnaie électronique "efficace". L'idée que Cecile Delerablée, Aline Gouget, Emeline Hufschmitt, Fabien Laguillaumie, Hervé Sibert, Jacques Traoré, Damien Vergnaud et moi-même avons eu [CDG+09] repose sur les techniques de "traitement par lots" (*batch* en anglais). Celles-ci avaient déjà été utilisées dans le domaine de la monnaie électronique mais essentiellement pour accélérer la vérification de la validité de plusieurs pièces déposées. Notre étude s'est portée sur la signature avec traitement par lots qui, dans le cas particulier de RSA, a été décrite par Fiat dans [Fia97].

Le principe du *batch RSA* consiste à obtenir, pour un unique module RSA, la signature

de plusieurs messages, chacun sur des exposants différents, en une seule exponentiation modulaire. Plus précisément, imaginons que nous ayons à notre disposition un module RSA n et, par exemple, 5 exposants publics RSA $e_0, \ldots, e_4$. Pour obtenir la signature de 5 messages distincts $m_0, \ldots, m_4$, après la compilation du message, il suffit d'une seule exponentiation modulaire avec l'exposant public $E = \prod_{i=0}^{4} e_i$. L'astuce proposée par Fiat consiste maintenant, à partir de cette unique exponentiation, à dériver les 5 signatures distinctes. En fait, il est même possible d'obtenir une signature sur n'importe quel succession de messages (par exemple, m_0, m_1 et m_2 d'un côté et m_3 et m_4 de l'autre, ou bien m_0, m_1, m_2 et m_3 puis m_4).

Dans le cadre de la monnaie, nous avons utilisé cette technique pour permettre à la banque de signer en une seule fois plusieurs pièces de monnaie, de telle sorte que chaque pièce puisse être dépensée indépendamment des autres, une par une ou même plusieurs à la fois. Les messages à signer correspondent aux numéros de série (par exemple S_0, S_1, S_2, S_3 et S_4) des pièces retirées. Ainsi, en une seule exponentiation, un utilisateur obtient plusieurs pièces signées par la banque. Il possède, à l'issue du retrait, une seule signature sur l'ensemble des pièces retirées (voir Figure 3.1). Par contre, il n'a pour l'instant pas besoin d'utiliser la technique de Fiat pour séparer cette unique signature en plusieurs : ceci sera fait lors de la dépense.

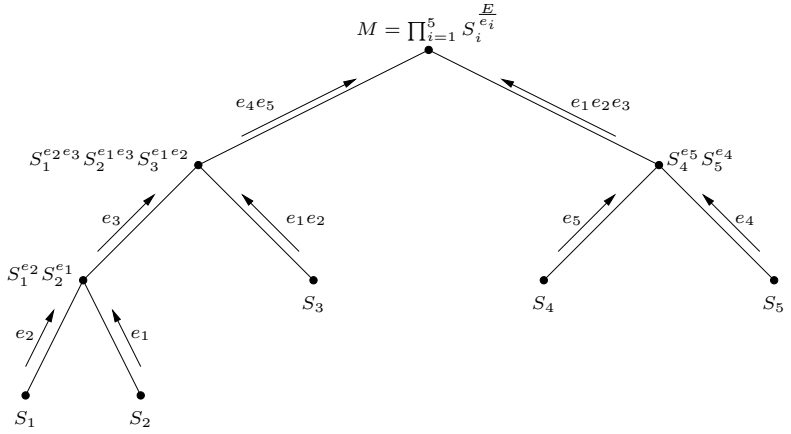

FIG. 3.1 – *Retrait de 5 pièces avec les numéros de série S_0 à S_4*

En effet, si un retrait a permis à un utilisateur d'obtenir par exemple 5 pièces, ce dernier pourra dépenser de la façon dont il le souhaite ces 5 pièces, par exemple 3 pièces, puis 2 autres, ou bien 4 pièces et puis la dernière. Pour cela, nous utilisons la technique de Fiat pour obtenir, à partir de la signature initiale sur l'ensemble des numéros de série, une signature sur les numéros de série des pièces uniquement dépensées, c'est-à-dire les 3 premières pièces dans un premier temps, puis les 2 dernières pour la seconde dépense dans le premier cas, ou bien les 4 premières et enfin la dernière dans le second cas.

Pour l'instant, nous n'avons pas encore assuré l'anonymat de l'utilisateur au moment de la dépense. Pour y parvenir, la signature par la banque lors du retrait doit être effectuée de telle sorte qu'il ne lui est pas possible de la reconnaître lorsqu'elle sera divulguée lors d'une dépense. Nous utilisons ici le principe de la signature aveugle, et plus particulièrement celle due à Chaum [Cha82] et basée sur la signature RSA, pour être compatible avec le batch RSA de Fiat.

D'un point de vue efficacité, nous obtenons quelque chose de très intéressant puisque le retrait permet d'obtenir plusieurs pièces en une seule signature de la part de la banque. Par ailleurs, la dépense ne met en œuvre que la décomposition de la signature initiale en deux parties :

1. celle contenant la signature des numéros de série des pièces dépensées et
2. celle contenant la signature des numéros de série des pièces encore à dépenser. Cette signature sera la base pour la prochaine dépense, c'est-à-dire que c'est elle qui sera décomposée par la même méthode, jusqu'à ce que l'utilisateur dépense ses dernières pièces, auquel cas il n'est pas utile de faire cette décomposition.

Cette étape de décomposition, décrite dans le papier initial de Fiat [Fia97] nécessite $\mathcal{O}(\log E)$ exponentiations modulaires.

Concernant la quantité de données qui transitent entre l'utilisateur et le marchand, l'envoi d'une unique signature est nécessaire mais il faut cependant, dans l'état actuel des choses, envoyer chacun des numéros de série des pièces dépensées. L'idée que nous avons eue au sein de PACE consiste alors à générer les numéros de série d'un même porte-monnaie de façon publique à partir d'une graine initiale et d'une fonction $\mathcal{F}$. En représentant tous ces numéros de série par un arbre, comme représenté par la Figure 3.2, cela permet à l'utilisateur de n'envoyer que le ou les nœuds de l'arbre correspondant aux pièces qu'il souhaite dépenser. Ainsi, s'il veut dépenser 4 pièces, il n'enverra au marchand que l'élément $s_{1,0}$. Par ailleurs, en utilisant la fonction $\mathcal{F}$, le marchand va être capable de générer à nouveau les numéros de série (*e.g.* S_0 à S_2) pour ensuite vérifier la signature RSA sur chacun d'eux. Cela nous permet d'obtenir une complexité en espace qui soit logarithmique en le nombre de pièces dépensées, ce qui est le meilleur schéma actuel.

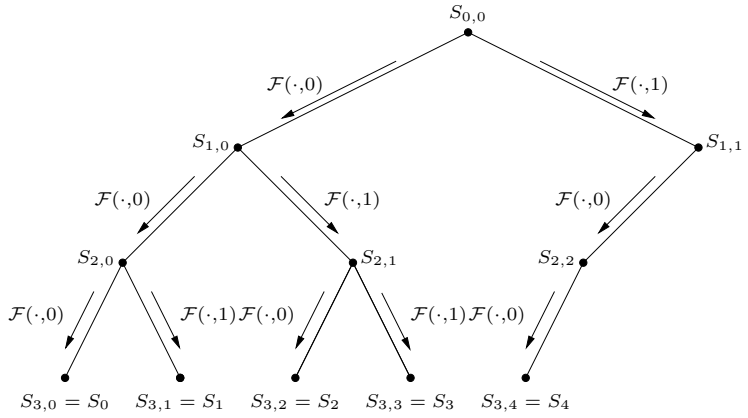

FIG. 3.2 – *Arbre binaire des numéros de série pour 5 pièces*

Il reste encore à résoudre le problème de la double dépense. En effet, une telle fraude va aisément être détectée puisqu'à la réception d'une signature sur plusieurs numéros de série, la banque va être capable, en utilisant la technique de Fiat, de générer les signatures RSA sur chacun des numéros de série. Si le numéro de série signé est déjà présent dans la base de données, alors il s'agit d'une double dépense. Dans ce cas, il n'y a pour l'instant aucun moyen de retrouver le coupable. Le problème est que nous utilisons une signature aveugle et que les techniques précédentes n'étaient pas très efficaces. Nous avons choisi une variante de celle

utilisée par Traoré [Tra99] en rajoutant, lors du retrait, une signature de type Camenisch-Lysyanskaya sur la graine des numéros de série et un secret u tel que $\mathsf{upk} = g_0^u$ est la clé publique de l'utilisateur. Lors de la dépense, il sera au final nécessaire de

1. révéler la signature aveugle sur les numéros de série dépensés, en utilisant la technique de Fiat ;
2. chiffrer la graine $s_{0,0}$ des numéros de série à l'aide de la clé publique du juge ;
3. chiffrer upk à l'aide de la clé publique du juge ;
4. prouver sa connaissance de la graine, de la clé secrète u et d'une signature Camenisch-Lysyanskaya de telle sorte que les chiffrés sont bien formés.

De façon assez surprenante, il n'est pas nécessaire de prouver le lien entre la graine $s_{0,0}$ et les numéros de série des pièces dépensées (e.g. S_0, S_1 et S_2). En fait, le chiffré de cette graine suffit pour retrouver le fraudeur le cas échéant. En effet, en cas de double dépense, il sera possible de déchiffrer la graine et de vérifier la validité du numéro de série. Si cela ne concorde pas, alors le chiffrement sur l'identité permettra de retrouver les identités des deux personnes impliquée et de trouver une explication.

Ce système n'est bien évidemment pas optimal et ce, pour deux raisons. Tout d'abord, la gestion des numéros de série entraîne un anonymat qui n'est plus aussi parfait puisque cela permet au marchand de savoir quelle partie d'une pièce retirée est en train d'être dépensée. Par ailleurs, nous nous plaçons dans le cadre de la monnaie équitable, qui est parfois contestée par les puristes de l'anonymat. Pourtant, je pense qu'il est possible d'améliorer ce système pour pallier ces problèmes et obtenir quelque chose de très intéressant.

3.2 La Monnaie Divisible

La monnaie divisible est aussi une réponse, certes partielle, à la dépense efficace. En quelques mots, cette technique consiste à retirer une “grosse” pièce de monnaie qu'il est par la suite possible de diviser en plusieurs “petites” pièces en fonction des dépenses à effectuer. Les premiers systèmes de monnaie divisible sont apparus dans les années 90. Ceux-ci n'atteignaient cependant pas un bon niveau d'anonymat puisqu'il était possible de détecter si deux “petites” pièces dépensées provenaient de la même “grosse” pièce ou non. Le premier système n'ayant pas cet inconvénient a été proposé dans [NS00]. Les auteurs se placent ici dans le cas de la monnaie équitable. Cette propriété additionnelle n'est pas anodine car elle simplifie énormément les choses du point de vue de l'anonymat. En effet, il suffit dans ce cas que le payeur chiffre son identité à l'aide de la clé publique de l'entité habilitée, puis prouve que ce chiffrement a correctement été effectué. Sans ce chiffrement, la difficulté est amplifiée puisqu'il ne doit être possible de lever l'anonymat qu'en cas de double-dépense. Par ailleurs, il est possible de savoir, dans ce système, quelle partie de la grosse pièce est en train d'être dépensée. L'anonymat fort n'est donc pas assuré, puisque si deux dépenses correspondent au même “morceau” d'une pièce dépensée, alors n'importe qui saura que ces deux dépenses proviennent de deux pièces différentes.

3.2.1 Une Méthode Générique pour un Anonymat Fort

Avec Aline Gouget, nous avons résolu une problématique ouverte depuis longtemps dans le monde de la monnaie électronique divisible. En effet, nous avons introduit, à la conférence Eurocrypt [CG07], le premier système de monnaie électronique divisible avec un anonymat total des utilisateurs, possédant donc les propriétés suivantes :

1. il n'est pas possible de savoir si deux dépenses proviennent de la même pièce retirée ou non ;
2. il n'est pas possible de savoir quelle partie de la pièce retirée est dépensée ;
3. il ne s'agit pas d'un système équitable.

La première propriété est obtenue de la même façon que pour le système de monnaie divisible proposé dans [NS00]. Le principe est d'utiliser un arbre pour représenter la pièce de monnaie. La racine représente la pièce dans sa totalité, et les feuilles correspondent (pour l'instant) aux pièces de valeur unitaire (par exemple 1 euro). Ainsi, pour retirer une pièce de valeur 2^ℓ euro, l'arbre de la pièce retirée aura une hauteur de ℓ. Chaque nœud a une valeur monétaire 2^i et les deux fils correspondants ont une valeur 2^{i+1}.

D'un point de vue représentation de la pièce, la racine est reliée à une clé, dénotée s, signée par la banque lors du retrait. Ensuite, pour un nœud donné, les clés des nœuds fils sont obtenues par l'application de deux fonctions à sens unique, une pour la clé du fils gauche et une autre pour la clé du fils de droite, sans qu'il y ait réellement de lien entre les deux fonctions. La dépense d'un nœud revient alors à révéler la clé du nœud dépensé. Comme il n'y a pas de lien entre les deux fonctions, il devient donc impossible de connaître le lien entre deux dépenses différentes : la première propriété est ainsi satisfaite.

Lors de la dépense d'une pièce, il est nécessaire de prouver que le nœud dépensé est correctement formé à partir du nœud racine qui a été signé par la banque. Il ne faut bien évidemment pas révéler le nœud racine. Mais, dans [NS00], les auteurs ont de plus révélé le chemin qui relie le nœud racine au nœud dépensé, révélant ainsi le sous-arbre, et donc la partie de la pièce, dépensé(e). Dans [CG07] nous avons proposé de cacher ce chemin en utilisant une preuve du "ou" telle que décrite dans la Section 1.1. Plus tard, il a aussi été proposé d'utiliser un accumulateur sur tous les nœuds d'une même ligne de l'arbre [ASM08] et de l'arbre entier [CG10], mais nous reviendrons sur ces schémas (et surtout le second) un peu plus tard.

L'obtention de la dernière propriété constitue notre principal apport dans le domaine de la monnaie divisible. Notre idée générale consiste à rajouter un niveau à l'arbre de la pièce, les feuilles devenant de ce fait des feuilles "mortes" sans valeur monétaire (voir Figure 3.3). La génération de l'arbre et des clés des nœuds reste par ailleurs inchangée par rapport à précédemment. A partir de là, il reste à définir la façon d'écrire le numéro de série et le tag de sécurité pour une pièce dépensée.

Notre solution fonctionne de la façon suivante :

1. le numéro de série est égale à la concaténation des deux clés des nœuds fils du nœud dépensé. Par exemple, si un utilisateur dépense le nœud situé en position (2,0) pour une valeur de pièce de 2 euro, le numéro de série $S = k_{3,0} \| k_{3,1}$ et
2. le tag de sécurité correspond à un chiffrement de l'identité upk du payeur avec la clé du nœud dépensé. Par exemple, $T = \text{Enc}(k_{2,0}, \mathsf{upk})$ avec l'exemple précédent et où Enc

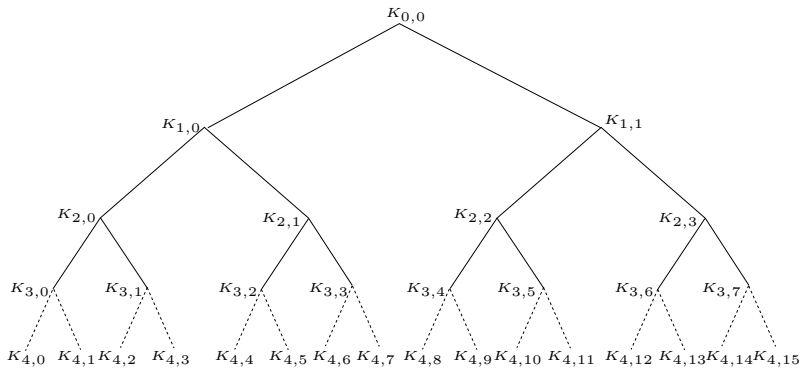

FIG. 3.3 – *Principe général - Arbre des clés*

sera détaillé un peu plus loin dans ce document.

Il est alors nécessaire de prouver la validité de cette dépense par rapport à un retrait valide, c'est-à-dire une signature de la banque sur certains éléments utilisés lors de cette dépense. Nous obtenons ainsi une méthode générique permettant d'obtenir un anonymat fort [1] pour la monnaie divisible. Cette méthode a été reprise dans toutes les constructions proposées [ASM08, CG10] depuis notre article.

3.2.2 Une Première Tentative de Mise en Œuvre Pratique

Dans [CG07], notre mise en œuvre pratique de la solution générique ci-dessus consiste à faire signer par la banque, lors du retrait, le secret s de la racine ainsi que la clé secrète u de l'utilisateur, reliée à la clé publique $\mathsf{upk} = g_0^u$. Lors de la dépense, il est alors nécessaire de prouver que le numéro de série et la clé utilisée pour chiffrer upk (voir l'étape 2 ci-dessus) sont reliés à s, et que upk est correctement liée à l'élément u signé. Le premier point est réalisé en prouvant que le chemin allant de la clé de la racine à celle du nœud étudié est bien formé, de proche en proche. Cependant, ceci rend le système relativement inefficace puisque la dépense va nécessairement dépendre de la valeur monétaire dépensée. En effet, dans notre cas, nous obtenons la clé k_{fils} du nœud fils à partir de celle k_{pere} du père en calculant $k_{fils} = g^{k_{pere}}$. Ainsi, la clé du nœud dépensé correspond, si on reprend notre précédent exemple, à la clé $k_{2,0} = g_2^{g_1^{g^s}}$. Il faut donc prouver, de proche en proche, que cette valeur est correctement construite à partir de s, ce qui nécessite l'utilisation de plusieurs preuves de connaissance d'un double logarithme discret, qui sont relativement inefficaces en pratique.

Par ailleurs, notre mise en œuvre pratique nécessite une génération des paramètres relativement difficile à réaliser efficacement. En effet, dans la génération des clés de l'arbre, il est nécessaire que k_{fils} et k_{pere} soient dans des groupes bien précis puisque le procédé de calcul des clés est réitéré plusieurs fois. Ainsi, pour retirer une pièce de valeur 2^ℓ, notre système nécessite $\ell + 2$ groupes $\mathbb{G}_0, \ldots, \mathbb{G}_{\ell+1}$ d'ordres premiers $p_0, \ldots, p_{\ell+1}$ respectivement, tels que pour tout $i \in [1,\ell+1]$, $\mathbb{G}_{i-1}$ est un sous groupe de $\mathbb{Z}_{p_i}$, c'est-à-dire $p_{i-1}|p_i - 1$. Nous avons ici repris la proposition de [NS00] en prenant $p_i = 2p_{i-1} + 1$ pour tout $i \in [1,\ell+1]$, ce qui

1. Attention, fort ne veut pas dire ici inconditionnel. Cela signifie qu'un adversaire n'apprend absolument rien sur la pièce dépensée.

correspond à une chaîne de Cunningham de longueur $\ell + 2$. Cependant, il n'existe pas de méthode pour générer une telle suite de nombres premiers plus efficacement qu'en procédant "brutalement". Dans ce cas, la probabilité de succès pour construire une telle suite est de 2^{-95} pour $\ell = 10$ et des éléments de 1024 bits !

Une proposition d'amélioration de cette efficacité a été faite dans [ASM08], en utilisant les accumulateurs, mais pas de façon optimale puisque cette proposition ne permet pas d'obtenir la propriété de non-falsification des pièces. En effet, lors du retrait, un utilisateur malhonnête possède une chance sur deux de retirer une pièce plus grosse que prévue. En utilisant la technique du couper-et-choisir (*cut-and-choose* en anglais), les auteurs obtiennent ainsi ce qu'ils appellent une "non-falsification statistique" puisque la banque a alors une chance sur deux de détecter un tel utilisateur frauduleux, et donc de le punir d'une amende. Ce procédé garantit qu'en "moyenne", la banque ne devrait pas perdre d'argent. Cette solution est difficilement acceptable en pratique.

3.2.3 Une Meilleure Efficacité

Dans [CG10], Aline Gouget et moi-même avons toutefois repris cette idée d'utiliser les accumulateurs pour construire un schéma de monnaie divisible avec la propriété d'anonymat fort, mais aussi avec une non-falsification "standard", tout en obtenant une dépense qui ne soit pas linéaire en la valeur monétaire de la pièce dépensée. Dans la méthode proposée dans [ASM08], les auteurs n'ont pas cherché à prouver le lien entre de la clé d'un nœud et celle des nœuds fils, obtenant ainsi une très bonne efficacité. Mais comme l'utilisateur n'a pas besoin de prouver que son arbre est bien formé à partir de la racine lors de la dépense, il peut tricher au moment du retrait et produire des arbres où les nœuds n'ont pas de lien les uns entre les autres, d'où la fraude.

Notre idée consiste à combiner notre premier article et la méthode des accumulateurs énoncée ci-dessus. Pour cela, lors du retrait, l'utilisateur va générer un arbre selon la même méthode que dans notre premier article, mais avec simplement une chaîne de Cunningham de longueur 2, ce qui est largement faisable en pratique. En effet, à chaque fois, nous allons ramener la clé du nœud fils dans un unique groupe $\mathbb{Z}_p^*$ par $k_{fils} = g^{k_{pere} \pmod q} \pmod p$ où q est l'ordre de g et où g est pris égal à g_0 (resp. g_1) s'il s'agit du fils gauche (resp. droit). De cette façon, il n'est plus possible de prouver que la clé d'un nœud donné est bien formée par rapport à la clé racine. Il est en revanche possible, contrairement au système de [ASM08], de prouver que deux nœuds donnés sont bien père et fils. Pour nous permettre de prouver la validité d'une dépense par rapport à un retrait valide, nous allons accumuler les clés des nœuds de l'arbre dans des accumulateurs. Ces accumulateurs seront par ailleurs signés par la banque lors du retrait, à l'aide d'une signature de type Camenisch-Lysyanskaya, en compagnie de la valeur u telle que $\mathsf{upk} = G^u$ est la clé publique de l'utilisateur, et d'un secret s calculé conjointement par la banque et uniquement connu par l'utilisateur. Au final, le retrait comporte les phases suivantes.

1. L'utilisateur génère l'arbre à partir d'une graine initiale $k_{0,0}$ et en calculant les clés de tous les nœuds[2] ;

2. Ce qui est différent de notre précédent système où il n'était pas nécessaire pour l'utilisateur de tout générer lors du retrait.

2. il accumule l'ensemble des clés des nœuds (exceptée celle de la racine) dans un unique accumulateur Acc ;
3. il crée $L+1$ accumulateurs supplémentaires $\mathsf{Acc}_1,\ldots,\mathsf{Acc}_{L+1}$ tels que l'accumulateur Acc_i accumule l'ensemble des clés au niveau i ;
4. il interagit avec la banque pour obtenir :
   - une signature de type Camenisch-Lysyanskaya sur l'accumulateur Acc, la valeur u et un secret s calculé conjointement par la banque et l'utilisateur ;
   - $L+1$ signatures $\sigma_1,\ldots,\sigma_{L+1}$ de type Camenisch-Lysyanskaya telles que la signature σ_i porte sur Acc_i, le secret s et l'élément i.

Lors de la dépense, notre solution consiste alors, comme dans notre système général d'Eurocrypt, à révéler les clés des nœuds des deux fils du nœud dépensé (par exemple $k_{3,0}\|k_{3,1}$), et à chiffrer la clé du nœud dépensé. Mais cette fois-ci, la preuve de validité de la dépense est effectuée en prouvant que toutes les clés des nœuds en dessous du nœud dépensé (non inclus) ont toutes été accumulées dans l'accumulateur signé par la banque lors du retrait et que les éléments du numéro de série sont bien accumulés dans un accumulateur signé par la banque au niveau dépensé. Il ne faut bien évidemment pas révéler ni les accumulateurs ni les signatures (puisque ces éléments sont uniques pour un retrait donné), mais par contre, les clés des nœuds sont connues du marchand puisque d'une part le numéro de série est révélé et d'autre part, le passage d'une clé père à une clé fils est public. Le chiffrement utilisé est ici très proche de celui que nous avions déjà proposé dans l'article d'Eurocrypt 2007, à savoir $T = \mathsf{upk} H^{R\cdot k_{2,0}}$, où R est un élément aléatoire non nul calculable par le payeur et le receveur et où H est un générateur public. Enfin, comme tous les éléments pour lesquels il faut prouver l'accumulation sont connus par le marchand, la preuve de validité correspondante ne dépend pas de la valeur de la pièce dépensée, ce qui est très intéressant d'un point de vue pratique. La preuve de validité de la pièce dépensée est donc au final :

$$\begin{aligned}
&\Pi = \textsc{Sok}\big(u,k_{L-\ell,j_0},s,\mathsf{Acc},\sigma,\mathsf{Acc}_{L-\ell+1},\sigma_{L-\ell+1} :\\
&T = G^u\cdot(H^R)^{k_{L-\ell,j_0}} \wedge k_{L-\ell+1,2j_0} = g_0^{k_{L-\ell,j_0}} \wedge k_{L-\ell+1,2j_0+1} = g_1^{k_{L-\ell,j_0}} \wedge\\
&(k_{L-\ell+1,2j_0},k_{L-\ell+1,2j_0+1},\ldots,k_{L+1,2^{\ell+1}j_0},\ldots,k_{L+1,2^{\ell+1}(j_0+1)-1}) \in \mathsf{Acc} \wedge\\
&(k_{L-\ell+1,2j_0},k_{L-\ell+1,2j_0+1}) \in \mathsf{Acc}_{L-\ell+1} \wedge \sigma = \textsc{Sign}(\mathsf{Acc},u,s)\\
&\sigma_{L-\ell+1} = \textsc{Sign}(\mathsf{Acc}_{L-\ell+1},s,L-\ell+1)\big)(\mathsf{mpk}\|\mathit{info}\|R\|S\|T)
\end{aligned}$$

Plus précisément, cette preuve Π prouve que :

- le tag de sécurité T est correctement formé à partir de u, R et $k_{2,0}$;
- le numéro de série S est correctement formé à partir de $k_{2,0}$;
- tous les descendants de la clé $k_{2,0}$ dans l'arbre sont accumulés dans Acc, sachant que tous ces éléments sont connus ou calculables par le marchand ;
- les deux éléments du numéro de série $k_{3,0}$ et $k_{3,1}$ sont accumulés dans Acc_3 ;
- l'accumulateur Acc, la clé secrète usk et un élément s sont signés par la banque dans σ ;
- l'accumulateur Acc_3, l'élément s et la valeur 3 sont signés par la banque dans σ_3 ;

sans révéler u, $k_{2,0}$, Acc, Acc_3, σ ni σ_3.

Nous obtenons alors trois systèmes de monnaie électronique divisible qu'il est possible de comparer à l'aide du tableau ci-dessous (où $\textsc{Dev}(i)$ correspond au temps nécessaire pour développer un polynôme de degré i).

Schéma de monnaie divisible	Au *et al.* [ASM08]	Canard-Gouget [CG07]	Canard-Gouget [CG10]
Modèle	Non falsification "statistique"	Non falsification	
Calcul pour l'arbre binaire	$(2^{L+1}-2)$Exp	$(2^{L+2}-2)$Exp (pas nécessairement calculé)	$(2^{L+2}-2)$Exp
Taille de stockage de la pièce divisible	$(2^{L+1}-2)\|p\| + (L+1)$Sign $+(L+1)$Acc	$2\|p\| + (1)$Sign	$(2^{L+2}-2)\|p\| + (L+2)$Sign $+(L+2)$Acc
Complexité calculatoire du retrait	$(L+1)$Sign $+(L+1)$Acc	(1)Sign	$(L+2)$Sign $+(L+2)$Acc
Complexité calculatoire de la dépense 2^ℓ	Exp + pok(Sign +Acc + 2Exp)	$(i+3)$Exp + pok(Sign $+\mathcal{O}((L-\ell)t)$Exp)	Exp + Dev(2^ℓ)+ pok(2Sign + 2Acc + 3Exp)
Taille de transfert d'une dépense	$2\|p\|$ + pok(Sign +Acc + 2Exp)	$3\|p\|$ + pok(Sign $+\mathcal{O}((L-\ell)t)$Exp)	$2\|p\| + \|P\|$ + pok(2Sign +2Acc + 3Exp)

Tab. 3.1 – *Comparaison d'efficacité des systèmes de monnaie divisible*

Au final, j'ai proposé avec Aline Gouget les deux premiers schémas de monnaie électronique divisible totalement sûrs. Le second [CG10] est par ailleurs tout à fait pratique, même s'il n'atteint pas le niveau d'efficacité du système de Au, Susilo et Mu [ASM08]. Mais, comme j'ai pu le montrer précédemment, ce dernier ne possède pas la propriété de non-falsification standard.

3.3 Le Transfert de Pièces

Je change maintenant de problématique pour m'intéresser à la transférabilité des pièces, c'est-à-dire la possibilité pour l'entité recevant une pièce de la dépenser à nouveau auprès d'une autre entité. Cette propriété permet de ce fait de résoudre la problématique de rendu de monnaie, puisque le marchant est capable de dépenser ses propres pièces auprès d'un client. Il n'y a plus de notion de marchand dans ce contexte.

Il y a en fait peu d'articles sur le sujet. Cela est essentiellement dû à un résultat de Chaum et Pedersen [CP92]. Ces derniers ont en effet montré de façon théorique qu'une pièce transférée d'un utilisateur vers un autre doit nécessairement grossir en taille. De façon très informelle, ceci est dû au fait que tous les possesseurs successifs de la pièce peuvent potentiellement dépenser deux fois cette pièce et qu'il est nécessaire, dans ce cas, de retrouver l'identité de la personne ayant dépensé deux fois la même pièce. Par conséquent, à chaque dépense, le payeur doit y inclure une partie de son identité afin d'être retrouvé en cas de fraude. Pourtant, l'amélioration des débits réseaux et des capacités de stockage, ainsi que celle des techniques cryptographiques peut laisser à penser que le domaine mérite que l'on s'y intéresse. On constate d'ailleurs un regain d'intérêt sur la monnaie transférable ces dernières années.

Les premiers schémas étaient relativement inefficaces, ou proposaient un niveau d'anonymat très pauvre puisqu'il était possible de tracer un même utilisateur dans toutes ses dépenses. Le premier schéma intéressant, utilisant les signatures aveugles, a été proposé dans la thèse de van Antwerpen [vA90] et repris par Chaum et Pedersen dans [CP92]. Cependant, ce système possède un inconvénient non négligeable. En effet, il oblige le receveur d'une pièce à

préalablement interagir avec la banque afin de retirer une pièce "vide" sans valeur monétaire. Ainsi, la dépense d'une pièce correspond en fait à un échange entre une "vraie" pièce et cette pièce vide. En pratique, cela est relativement gênant puisqu'il rend le système, de mon point de vue, inintéressant. En effet, l'un des intérêts majeurs de la monnaie transférable est la diminution significative des interactions entre les différents acteurs (clients et marchants) et la banque, puisque la même pièce retirée va servir à plusieurs dépenses avant de revenir auprès de la banque, au moment du dépôt final. Le fait de devoir retirer cette pièce vide pour chaque pièce reçue nécessite d'interagir avec la banque pour chaque pièce reçue, et donc pour chaque dépense.

3.3.1 Une Nouvelle Proposition Efficace

Avec Aline Gouget et Jacques Traoré, nous avons cherché à résoudre le problème du système précédent en proposant un système de monnaie transférable [CGT08] sans cet inconvénient. Nous sommes ainsi les premiers à avoir proposé un tel système, montrant de ce fait que la recherche sur la monnaie transférable a un intérêt puisque des schémas quasiment pratiques existent.

Nous avons en fait proposé deux systèmes, l'un d'eux fournissant un anonymat inconditionnel des utilisateurs. Nous avons une nouvelle fois utilisé le principe de tag de sécurité de *compact e-cash*, mais il a bien évidemment fallu l'adapter. Il y a plusieurs problématiques essentielles à étudier dans le cas de la monnaie transférable :

1. il faut que la personne qui dépense la pièce donne les droits de dépense de cette pièce au receveur (et uniquement à lui). Il ne faut par ailleurs pas que quelqu'un écoutant la conversation liée à la dépense puisse récupérer ce droit. De ce fait, il faut que la dépense d'une pièce contienne une donnée que seul le receveur sera capable d'utiliser lorsqu'il dépensera la pièce reçue. Dans [vA90, CP92], cette propriété était obtenue à l'aide de la pièce vide. Nous avons choisi une autre approche qui consiste à inscrire chaque utilisateur une et une seule fois auprès d'une autorité désignée (par exemple la banque, mais ce n'est pas nécessaire). Lors de cette inscription, les utilisateurs vont obtenir une signature sur une donnée secrète u leur appartenant. Cette donnée secrète devra être utilisée par le receveur pour calculer une donnée pseudo-aléatoire qui devra nécessairement être utilisée par le payeur. Lorsque le receveur dépensera à nouveau cette pièce, il devra prouver sa connaissance de la donnée secrète et de la signature correspondante, prouvant ainsi qu'il a le droit de dépenser la pièce en jeu;
2. une double dépense doit être détectée par la banque au moment du dépôt final. Nous avons repris ici le principe décrit par van Antwerpen. Ainsi, une double-dépense est effective lorsqu'une pièce avec le même numéro de série et le même niveau de dépense est détectée, comme décrit par la Figure 3.4.
   Ainsi, chaque nouvelle dépense va inclure le numéro de série ainsi que tous les éléments permettant de vérifier les dépenses précédentes de cette même pièce. La banque sera alors capable de vérifier la totalité des dépenses successives de cette pièce et de détecter ou a eu lieu la double-dépense, le cas échéant ;
3. en cas de double dépense d'une pièce, il faut être capable de remonter au fraudeur, mais uniquement à lui et à aucun autre payeur de la même pièce. De façon similaire, un utilisateur ne doit pas être faussement accusé d'avoir double dépensé une pièce alors

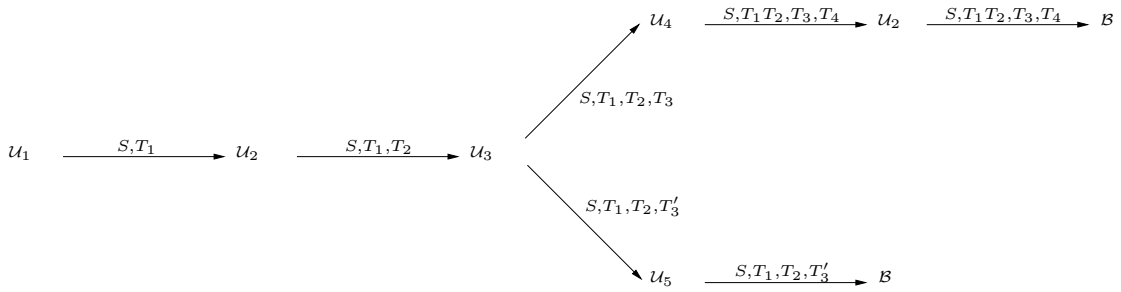

FIG. 3.4 – *Détection d'une double dépense en monnaie transférable*

que ce n'est pas le cas. Cela doit inclure les deux cas de figure suivants :

- un utilisateur dépense deux pièces différentes. Le tag de sécurité doit donc inclure un élément propre de la pièce. Cet élément ne peut pas provenir, comme c'est le cas habituellement pour le tag de sécurité (voir l'élément t utilisé au dessus), du retrait de la pièce dépensée puisque le payeur ne connaît pas forcément (et ne doit pas connaître) les éléments secrets provenant du retrait initial de la pièce qu'il dépense ;
- un utilisateur peut éventuellement obtenir deux fois la même pièce à deux moments différents, et donc la dépenser deux fois. Dans ce cas, ce n'est bien évidemment pas une fraude et il ne faut pas que l'identité de l'utilisateur en question soit retrouvée.

Le numéro de série d'une pièce va donc être généré tel que dans le système *compact e-cash*, c'est à dire par $S = g^{\frac{1}{s+j+1}}$ où s a été signé par la banque lors du retrait. Il sera présent dans toutes les dépenses de cette pièce, de façon inchangée (voir Figure 3.4).

Le tag de sécurité d'une pièce qui vient d'être retirée sera de nouveau identique à celui de *compact e-cash*, à savoir $T = \mathsf{upk} \cdot h^{\frac{R}{t+j+1}}$ mais avec une valeur R différente. En effet, nous allons nous appuyer sur cette valeur pour permettre au receveur de dépenser à nouveau la pièce reçue (voir le point 1. ci-dessus). Pour cela, R va être calculée par $\mathcal{H}(r\|d\|d')$ où d (resp. d') est une valeur choisie par le receveur (resp. payeur) où $r = g_0^{\frac{1}{\tilde{u}+d}}$ avec $\tilde{u}$ la clé secrète du receveur, liée à la clé publique $\widetilde{\mathsf{upk}} = g_0^{\tilde{u}}$ et signée par la banque (ou une autre autorité) avec une valeur notée σ.

Lorsque la pièce sera à nouveau dépensée, le tag de sécurité deviendra alors $T = \widetilde{\mathsf{upk}} \cdot h^{\frac{\tilde{R}}{S+\tilde{u}+h}}$ où $\widetilde{\mathsf{upk}}$ et $\tilde{u}$ appartiennent au payeur, S est le numéro de série de la pièce et h est l'historique des dépenses de cette pièces (par exemple $h = \mathcal{H}(S\|\hat{T}\|\bar{T}\|T)$). Ce dernier est utilisé pour permettre à un même utilisateur de recevoir deux fois la même pièce à deux moments différents, sans être accusé de fraude. La preuve de validité de cette dépense correspond à une signature de connaissance (via l'heuristique de Fiat-Shamir) qui comprend la preuve que T est bien formé, que le payeur possède un certificat sur la clé publique upk utilisée et surtout une preuve de bonne formation du r utilisé lors de la dépense précédente, prouvant de ce fait qu'il possède les droits de dépenser cette pièce :

$$\tilde{V} = \textsc{Pok}(\tilde{u},\tilde{\sigma} : T = g_0^{\tilde{u}} h^{\frac{R}{\tilde{u}+S+h}} \wedge r_l = g_0^{\frac{1}{\tilde{u}+d'}} \wedge \tilde{\sigma} = \textsc{Sign}(\tilde{u}))$$

La dépense correspond donc maintenant à $(S,T,V,\tilde{T},\tilde{V})$ plus tous les éléments d et r successifs

qui sont utiles pour vérifier que cette pièce dépensée est valide, où V est la preuve de validité de la dépense initiale.

Les autres procédures se décrivent maintenant aisément avec les explications ci-dessus. Notamment, une double dépense est aisément détectable par la banque qui va obtenir deux fois le même numéro de série S (il ne peut pas être modifié par un utilisateur puisque sa preuve de validité est dans la première dépense) et, en cas de double dépense d'une même pièce, les deux tags de sécurité vont inclure

- le même numéro de série S puisque c'est la même pièce qui est dépensée deux fois ;
- le même secret $\tilde{u}$ puisqu'il faut en prouver sa connaissance avec la signature $\tilde{\sigma}$;
- le même historique h puisqu'il s'agit d'une double dépense et pas d'avoir dépensé deux fois la même pièce à deux niveaux différents (voir le point 3 ci-dessus) ;
- deux R différents puisque les marchands sont différents.

Il sera donc possible de retrouver $\widetilde{\mathsf{upk}}$, en utilisant les techniques classiques.

Dans [CGT08], nous avons aussi proposé une variante de ce schéma proposant un anonymat inconditionnel. Je ne décrirai pas ce second système et le lecteur intéressé pourra se référer à l'article en question.

3.3.2 Problématique d'Anonymat dans la Monnaie Transférable

On définit habituellement deux niveaux d'anonymat pour la monnaie électronique. Ainsi:

- un système vérifie la propriété d'**anonymat faible** s'il n'est pas faisable de savoir si une dépense et un retrait sont liés. Il est par contre possible de savoir si deux dépenses ont été effectuées par le même utilisateur ou non ;
- un système vérifie la propriété d'**anonymat fort** s'il n'est pas faisable de savoir si deux dépenses ont été effectuées par le même utilisateur ou non. Un adversaire est par contre capable de reconnaître une pièce qu'il a déjà vue précédemment.

Dans le schéma que j'ai décrit ci-dessus, comme d'ailleurs dans tous les schémas de monnaie transférable existants, un adversaire est toujours capable de reconnaître une pièce qu'il a déjà vue précédemment. En effet, comme le numéro de série unique de la pièce transite dans toutes les dépenses de cette pièce (cf. Figure 3.4), cet adversaire est capable de fournir une unique pièce à un utilisateur pour être ensuite capable de le tracer lors de sa prochaine dépense.

Avec Aline Gouget [CG08], nous avons mis en place un nouveau modèle de sécurité propre à la monnaie transférable qui prend en compte l'attaque ci-dessus et définissant de ce fait de nouvelles propriétés d'anonymat. Plus précisément, nous en avons rajouté deux nouvelles qui sont informellement définies de la façon suivante :

- un système vérifie la propriété d'**anonymat complet** s'il n'est pas faisable de reconnaître une pièce qu'on a déjà vue précédemment. Un adversaire est par contre capable de savoir si une pièce qu'il reçoit a déjà été entre ses mains ;
- un système vérifie la propriété d'**anonymat parfait** s'il n'est pas faisable de savoir si une pièce que l'on reçoit a déjà été en notre possession.

Les définitions plus formelles sont présentes dans [CG08]. Il est par contre clair que :

$$\text{anonymat parfait} \Longrightarrow \text{anonymat complet} \Longrightarrow \text{anonymat fort} \Longrightarrow \text{anonymat faible.}$$

Les anonymats faible et fort ont déjà été atteints par les schémas existants. Il est par exemple possible de montrer formellement que le schéma de van Antwerpen et ceux que j'ai proposés avec Aline Gouget et Jacques Traoré (décrits ci-dessus) vérifient la propriété d'anonymat fort (et donc celle d'anonymat faible). Par contre, aucun d'eux, à ma connaissance, n'atteint la propriété d'anonymat complet (et a fortiori celle d'anonymat parfait).

Dans [CG08], nous avons décrit un schéma générique permettant, à partir d'un schéma ayant la propriété d'anonymat fort, d'obtenir un schéma possédant un anonymat complet. Comme l'adversaire n'est pas actif lors de ces phases (son but consiste simplement à observer des dépenses et à dire lesquelles sont liées à la même pièce), notre proposition consiste donc en la sécurisation des données qui transitent entre le payeur et le receveur. Pour cela, il suffit de chiffrer les communications lors de la dépense, en utilisant une clé de session. Ainsi, préalablement à tout échange, le payeur et le receveur doivent s'échanger cette clé de façon sécurisée.

Concernant l'anonymat parfait, Chaum et Pedersen ont montré dans [CP92] qu'elle ne pouvait pas être atteinte face à un adversaire tout puissant (*i.e.* disposant d'une puissance de calcul non polynomialement bornée). Dans [CG08], nous avons montré que, si l'on souhaite atteindre cette propriété, la procédure de dépôt doit nécessairement être secrète et que ce secret (par exemple l'accès à la base de données de toutes les pièces déposées) ne doit pas être connu par l'adversaire contre l'anonymat parfait. En particulier, la banque (ou plus précisément l'entité en charge du dépôt) va nécessairement gagner l'expérience liée à ce niveau d'anonymat. Nous avons donc défini une nouvelle propriété, n'ayant aucun lien avec les autres, nommé **anonymat parfait secondaire**, où l'adversaire ne possède pas cet élément secret.

Cette nouvelle propriété ne semble pas facile à obtenir puisqu'il faut que les données des précédentes dépenses transitent dans la nouvelle dépense, et notamment le tag de sécurité. En effet, en cas de fraude détectée au moment du dépôt, il faut retrouver l'identité du fraudeur, où qu'il soit dans les deux chaînes de dépense. Dans notre papier, nous n'avons pas réussi à construire un tel schéma de façon efficace. Par contre, nous avons montré que cette propriété était atteignable en exhibant un schéma théorique prouvé sûr : il est donc possible de mettre en place un système de monnaie transférable garantissant qu'une entité recevant une pièce sera incapable de savoir si elle l'a déjà eue entre ces mains.

Pour le numéro de série et le tag de sécurité, il est par exemple possible de les chiffrer lors de la première dépense, puis pour les dépenses suivantes, de les rechiffrer ou de les rerandomiser à l'aide d'un schéma de chiffrement vérifiable pour être capable de prouver que la valeur chiffrée possède les bonnes propriétés (et donc que la banque sera capable de retrouver le fraudeur en cas de double dépense). On obtient ainsi la chaîne suivante :

$$(S,T) \longrightarrow (dS = \textsc{Enc}(S), dT = \textsc{Enc}(T)) \longrightarrow (d^2S = \textsc{Enc}(dS), d^2T = \textsc{Enc}(dT)) \longrightarrow \dots$$

A chaque dépense, il va donc falloir rechiffrer le numéro de série et l'ensemble des tags de sécurité présents dans l'historique des dépenses de cette pièce, puis rajouter son propre tag de sécurité. Il reste encore à prouver que la dépense effectuée est valide et qu'elle provient bien d'un retrait valide. L'idée consiste alors à transférer la preuve de validité des dépenses précédentes, en la modifiant pour qu'elle ne puisse pas être reconnue par une personne l'ayant déjà eue entre les mains. Nous avons d'abord souhaité effectuer une preuve de connaissance d'une preuve de connaissance vérifiant les bonnes propriétés, sans révéler la preuve initiale. Mais cela ne semble pas possible de réaliser une telle preuve avec les systèmes actuels, notam-

ment à cause de l'oracle aléatoire. Notre choix s'est alors porté sur une construction générique non efficace, utilisant les "méta preuves", concept de preuve de connaissance d'une preuve de connaissance introduit dans [SY90]. Chaque entité dépensant une pièce va alors prouver qu'elle connaît une preuve de validité de la dépense précédente, mais sans la révéler.

3.4 Conclusion et Perspectives

C'est sans doute dans le domaine de la monnaie électronique que j'ai obtenu mes résultats les plus marquants, sur des problématiques ouvertes depuis de nombreuses années. Je suis ainsi le premier à avoir proposé un système de monnaie divisible avec un niveau d'anonymat aussi fort et un système de monnaie transférable n'obligeant pas un utilisateur à interagir avec la banque avant de recevoir une pièce. J'ai ainsi favorisé la relance de la recherche cryptographique dans ce domaine.

La dépense efficace de plusieurs pièces de monnaie n'est pas encore tout à fait résolue. Les systèmes basés sur le système *compact e-cash* et les travaux que nous avons effectués dans le cadre du projet PACE apportent une nouvelle vision, mais l'efficacité et/ou l'anonymat obtenus ne sont pas optimaux. Une première approche serait d'étudier encore plus en détails ces techniques, afin de trouver la façon de dépenser efficacement. Une autre possibilité serait de changer la façon de concevoir la monnaie électronique, en se basant sur des techniques cryptographiques différentes. Je ne suis pas convaincu que l'une de ces méthodes soit bien meilleure que l'autre, alors il me semble important de mener les deux de front.

Concernant la problématique de transférabilité des pièces, l'apparition récente des preuves de validité de Groth et Sahai [GS08] dans des environnements bilinéaires permet d'envisager une autre approche qui consiste à re-randomiser la preuve initialement reçue et ainsi, améliorer grandement l'efficacité. Cependant, il reste encore des perspectives de recherche puisqu'il est maintenant nécessaire d'adapter le tag de sécurité, ce qui ne semble pas immédiat.

Il est aussi primordial de s'intéresser au problème du grossissement de la pièce, transfert après transfert. Récemment, Fuchsbauer *et al.* [FPV09] ont par exemple proposé que chaque utilisateur conserve une partie de la pièce dépensée. Celle-ci lui sera demandée, en cas de double dépense, pour qu'il prouve que ce n'est pas lui le fraudeur. Cette solution n'est pas optimale car très contraignante pour les utilisateurs[3]. Elle a par contre le mérite de proposer une nouvelle vision des choses qui me paraît très prometteuse.

Il reste certainement de nombreuses nouvelles propriétés qui pourraient être rajoutées aux systèmes de monnaie électronique existants. On pourrait par exemple citer la dépense anonyme ou non avec le même porte-monnaie, ou bien le problème de la taille de la base de données de la banque, mais il y en a certainement d'autres. La nouvelle directive en étude au niveau européen et permettant l'entrée sur le marché de la monnaie électronique d'établissements non nécessairement bancaires va certainement dynamiser cette recherche de nouveaux services.

3. Ils doivent en effet stocker des données pour chaque transaction qu'ils effectuent, ce qui à terme peut s'avérer trop important pour la capacité de stockage des utilisateurs.

Chapitre 4

La Cryptographie (Ultra) Légère

DAns ce chapitre, je vais aborder les problématiques de protection de la vie privée, utilisant bien évidemment de la cryptographie mais, cette fois-ci, l'entité qui effectue les calculs est un dispositif limité en espace mémoire et en capacité de calculs. J'ai déjà eu l'occasion de présenter des résultats sur ce sujet lors de ma thèse [CG02]. J'ai par la suite continué mes travaux, soit en étudiant d'autres briques de base [CT04], soit en m'intéressant plus particulièrement à des services [CT04, BCG05, CS06b, CS06a]. Ces études ont montré l'étendue du travail à effectuer et c'est pour cela qu'avec Marc Girault, nous avons souhaité encadrer ensemble une thèse, celle d'Iwen Coisel, qui a commencé en 2006. Elle a donné lieu à deux publications et à deux soumissions en cours, dans le domaine des cartes à puce [CC10] mais surtout dans celui de la protection de la vie privée dans les systèmes RFID [CC08, CCE10, CCG10].

Sommaire

La cryptographie légère, aussi appelée cryptographie à bas coût, est un domaine de recherche assez récent, et qui a vu un essor très important, notamment depuis que les cryptologues s'intéressent au monde des étiquettes RFID.

Dans les services de télécommunications, ce domaine englobe l'utilisation des cartes à puce mais aussi, et surtout, celui des cartes SIM. Dorénavant, la plupart des gens possèdent un téléphone mobile embarquant une carte SIM et on peut imaginer, comme c'est déjà le cas dans certains pays asiatiques ou plus largement avec le phénomène iPhone, que le téléphone va progressivement être utilisé pour des applications diverses. Cependant, ces dispositifs ne possèdent pas la puissance d'un ordinateur. Or, le nombre de calculs cryptographiques qu'il est nécessaire d'effectuer pour réaliser un système de vote électronique (voir Section 2.1), de

fédération d'identité avec protection de la vie privée des utilisateurs (voir Section 2.2) ou de monnaie électronique (voir le Chapitre 3) est tellement grand qu'on peut raisonnablement se demande si la transposition de ces services dans le monde mobile, et donc des briques cryptographiques dans les cartes à puce (y compris cartes SIM), est possible ou non.

Le constat est encore plus vrai dans le cas des systèmes de RFID puisque les étiquettes sont, pour les moins puissantes, à peine capables de réaliser une multiplication (non modulaire) ou un calcul de fonction de hachage. Dans ces cas là, il devient primordial de créer des algorithmes "sur mesure".

L'expérience montre qu'il faut adapter les algorithmes et/ou faire des hypothèses supplémentaires,. C'est ce que j'ai fait dans le cadre de mes travaux de recherche, comme je vais le montrer dans ce chapitre.

4.1 Cartes à Puce dans les Services

Lors de ma thèse, je m'étais intéressé à l'utilisation de l'hypothèse d'inviolabilité d'une carte à puce pour la création de mécanismes cryptographiques complexes. J'ai depuis repris cette idée pour certaines constructions cryptographiques [CT04], ou dans le cas de certains services [CT04, BCG05, CS06b, CS06a]. Une autre idée, appliquée dans les travaux de thèse d'Iwen Coisel, consiste à aider la carte à puce en lui donnant moins de calculs à faire, le reste étant effectué par une entité dite intermédiaire.

4.1.1 Hypothèse d'Inviolabilité des Cartes

Mes travaux de recherche pendant ma thèse avaient également porté sur les signatures de groupe et ses applications. Pour cela, j'avais notamment remarqué qu'il était nécessaire, pour certains services, d'adapter cette brique cryptographique en faisant des modifications dans les propriétés, obtenant ainsi, par exemple, les signatures de liste. J'ai donc aussi traité l'implémentation des signatures de groupe dans les cartes à puce [CG02], en faisant l'hypothèse que le dispositif ne peut pas être attaqué, c'est-à-dire que (i) les clés embarquées sont en sécurité et que (ii) les algorithmes embarquées dans la carte s'exécutent tout le temps de façon normale. Cette étude a par la suite été reprise pour les signatures d'anneau dans [XY04] et j'ai fait de même avec Jacques Traoré, après ma thèse, justement pour les signatures de liste mais aussi pour d'autres variantes [CT04].

Dans le système que j'ai introduit, l'appartenance au groupe va se caractériser par la possession d'une carte à puce étant capable de produire une signature électronique (classique) $\textsc{Sign}$ sur un message m. L'anonymat au sein du groupe sera obtenu en donnant à chaque carte toujours la même clé de signature gsk. Ceci est possible et ne pose pas de problème de sécurité si nous nous plaçons sous l'hypothèse d'inviolabilité des cartes à puce puisqu'il n'est dans ce cas pas possible de compromettre l'intégrité d'une carte pour y récupérer les clés embarquées, et donc cette clé de signature :

$$\sigma = \textsc{Sign}(\mathsf{gsk},m).$$

Dans le cas des signatures de groupe, une entité habilitée doit être capable, le cas échéant, de lever l'anonymat d'une signature. Dans [CG02], nous avions donc rajouté à cette signature

de groupe le chiffrement (probabiliste) Enc d'un identifiant uid propre à la carte à l'aide de la clé apk du manageur de révocation d'anonymat :

$$c = \text{Enc}(\mathsf{apk},\mathsf{uid}).$$

L'anonymat est ainsi conservé, sauf pour l'autorité habilitée, ce qui est conforme aux définitions données dans la Section 1.3.

Pour les signatures de liste, il n'est pas nécessaire de lever l'anonymat ni d'avoir un chiffrement. Par contre, il faut pouvoir relier plusieurs signatures d'une même carte lors d'une même période (voir la Section 1.3 pour des précisions sur la notion de période). Pour cela, nous avons utilisé un générateur pseudo-aléatoire Prg prenant en entrée une clé secrète usk propre à la carte et un élément rpe représentant la période en cours (et étant donc modifié à chaque période) :

$$r = \text{Prg}(\mathsf{usk},\mathsf{rpe}).$$

Pour un utilisateur et une période donnés, la sortie r sera en effet tout le temps la même et on obtient ainsi bien la propriété de traçabilité partielle.

Il est enfin possible de limiter le nombre de signatures anonymes qu'une entité doit être capable d'effectuer, en partie dans le but d'atteindre les signatures "tramway" qui m'avaient été demandées par Jacques Stern lors de ma soutenance de thèse, autrement appelées les "attestations anonymes k-fois" (les k-*TAA* en anglais). Avec l'hypothèse d'inviolabilité des cartes, cette propriété supplémentaire s'obtient en faisant confiance à l'exécution de l'algorithme embarqué dans la carte, qui va nécessairement s'arrêter de signer après k exécutions pour une période donnée rpe.

Ainsi, chacune des briques de base (signature, chiffrement, générateur pseudo-aléatoire et restriction sur le nombre de signatures) permet d'obtenir une propriété différente (anonymat, levée d'anonymat et traçabilité partielle, limitée ou non) pour la brique cryptographique globale. Autant la signature est nécessaire, puisque nous nous intéressons à l'anonymat dans les services, autant les autres propriétés ne sont pas tout le temps utiles. Il est donc possible de d'assembler ces briques comme on le souhaite, selon les besoins du service.

La seule restriction est qu'il faut que les éléments c et r soient, le cas échéant, signés par la procédure Sign, au même titre que le message. Dans tous les cas, le résultat est très efficace, surtout par rapport aux schémas initiaux ne faisant pas l'hypothèse d'inviolabilité de la carte à puce. En effet, dans le pire des cas, la production d'une signature ne demande qu'une signature classique (par exemple RSA, ce qui est dorénavant très rapide avec un cryptoprocesseur), un chiffrement (à nouveau RSA) et un calcul d'entier aléatoire par un générateur pseudo aléatoire (utilisant par exemple un générateur basé sur une fonction de hachage). Dans les schémas "classiques", il faut au minimum une dizaine d'exponentiations modulaires (soit dix fois plus longtemps) pour obtenir le même résultat (voir le Chapitre 1).

J'ai ainsi montré par mes travaux qu'il était possible d'obtenir des propriétés très complexes à l'aide de l'hypothèse d'inviolabilité des cartes et d'outils cryptographiques de base.

4.1.2 Utilisation dans les Services

A l'aide des briques ci-dessus, comme dans le Chapitre 2, il est ensuite possible de spécifier les briques de sécurité de certains services pour lesquelles la protection de la vie privée des utili-

sateurs est souhaitable. Cette fois-ci, je me suis placé dans le cas d'un utilisateur possédant un dispositif ayant très peu de ressources, comme par exemple une carte à puce ou un téléphone mobile muni d'une carte SIM. Comme je l'ai montré avec Jacques Traoré dans [CT04], il est possible d'utiliser les briques décrites ci-dessus pour des tickets d'abonnement anonymes ou des appels d'offres. Je ne vais pas détailler ici ces services et le lecteur intéressé pourra se référer à [CT04] pour les détails.

En revanche, voici l'étude que j'ai pu faire sur deux services en particulier.

Vote électronique. Dans le cas du vote électronique, tel que je l'ai déjà décrit lors de ma thèse [CT03a], les signatures de liste sont parfaitement adaptées. Lors d'une collaboration entre mon employeur et un fondeur de cartes à puce, Hervé Sibert et moi-même avons souhaité mettre en place une système de vote électronique sûr, avec le minimum d'opérations cryptographiques à effectuer par la carte à puce d'un électeur [CS06b, CS06a]. Nous avons donc imaginé un système complet de vote où chaque électeur doit se déplacer dans un bureau de vote, muni de sa carte à puce d'électeur.

Dans l'isoloir, l'électeur va effectuer son choix ch et la carte va produire une signature de liste LSign sur ce dernier, puis va chiffrer l'ensemble constitué de la signature et du choix pour obtenir le bulletin :

$$b = \text{Enc}(ch, \text{LSign}(ch)).$$

Dans une seconde phase, face à l'urne électronique, la carte va envoyer ce bulletin, ainsi que l'émargement de l'électeur. Ce dernier sera utilisé pour vérifier que l'électeur a effectivement voté. Les scrutateurs vont être capables de dépouiller les bulletins en les déchiffrant, puis en vérifiant leur validité à l'aide de la signature de liste.

Ce système a atteint son objectif : être très efficace et être conforme à ce qui peut être demandé à un véritable système opérationnel. Il a ainsi été implémenté dans un prototype incluant une véritable carte à puce.

Anonymisation. Dans le cas de la santé, et plus particulièrement du tiers payant chez un professionnel de santé, certaines mutuelles complémentaires souhaitent aujourd'hui fournir un service personnalisé à leurs clients. Pour autant, la loi interdit aux mutuelles d'obtenir les informations précises (appelées données détaillées) d'une prestation de santé. Par exemple, la mutuelle ne connaîtra pas l'identifiant exact d'un médicament acheté chez un pharmacien, mais obtiendra seulement une référence portant sur un groupe de médicaments (les données simplifiées). Il existe cependant deux cas particuliers :

1. soit l'utilisateur consent à ce que les données détaillées soient transmises à sa mutuelle. Il doit signer son consentement pour ce service de santé ;
2. soit l'utilisateur est anonyme.

J'ai eu l'occasion de proposer une solution pour répondre à cette seconde possibilité, dans le cas où chaque patient possède une carte à puce (l'équivalent de la carte de sécurité sociale pour les mutuelles). Dans le cas étudié ici, l'hypothèse d'inviolabilité n'était pas envisageable et la carte ne pouvait implémenter que les opérations standards de signature et de chiffrement (RSA). Une idée pourrait en effet être d'utiliser les signatures de groupe déléguées dans un

système proche de celui proposé dans la Section 2.4, mais ce dernier nécessite des briques cryptographiques spécifiques, ce qui n'était pas possible dans le cas étudié ici.

La solution que j'ai brevetée avec François Boudet et Stéphane Guilloteau [BCG05] consiste à introduire une nouvelle entité possédant les données détaillées mais pas l'identité des patients. A l'inverse, la mutuelle va connaître l'identité du patient mais n'obtiendra que les données simplifiées.

Ainsi, lors d'une prestation de santé, la carte à puce du patient va signer les données détaillées à l'aide d'une clé publique reliée à un identifiant Id_c connu à la fois de la mutuelle et de l'intermédiaire, puis va chiffrer cette signature et un identifiant NA uniquement connu de l'intermédiaire :

$$\sigma = \textsc{Sign}_{Id_c}(DD), c = \textsc{Enc}(NA, \sigma).$$

Cette dernière va déchiffrer la signature et l'identifiant et, à l'aide de ce dernier, va retrouver dans sa base l'identifiant commun (ce qui va lui permettre de vérifier la signature σ) ainsi que les droits de remboursement du patient. Ceux-ci vont pouvoir être adaptés en fonction de son contrat qui le lie à la mutuelle, l'entité intermédiaire ayant connaissance de façon anonyme de ce contrat. Elle va ensuite faire suivre à la mutuelle les données simplifiées et le remboursement effectué, permettant ainsi à cette dernière d'en informer le patient.

J'ai effectué ces recherches avec mes collègues, soit sur des demandes spécifiques, soit à partir de problématiques que nous nous sommes posées. Ces systèmes innovants sont très proches des besoins réels qui nécessitent l'utilisation d'une carte à puce disposant de très peu de ressources (mais sans nécessairement l'hypothèse d'inviolabilité de la carte). Pourtant, dans certains cas, l'hypothèse d'inviolabilité n'est pas possible et/ou les algorithmes standards ne sont plus suffisants. Il faut donc trouver d'autres solutions.

4.1.3 Cryptographie Anonyme Assistée

Si l'on souhaite créer un système de vote électronique où chaque électeur possède une carte à puce peu puissante mais sans vouloir faire l'hypothèse d'inviolabilité, comme c'est le cas avec les systèmes décrits ci-dessus, par exemple parce que l'on considère qu'elle est trop forte et peu réalisable en pratique, la problématique devient beaucoup plus complexe. Le constat est le même pour le principe de fédération d'identité présenté dans la Section 2.2, ou bien les systèmes de monnaie électronique présentés dans le chapitre précédent. Sans hypothèse d'inviolabilité, il devient très difficile, voire impossible, d'obtenir de tels systèmes.

Ces services nécessitent essentiellement l'utilisation des briques de base pour l'anonymat que j'ai pu présenter dans le Chapitre 1, à savoir les signatures de groupe, les signatures aveugles et les signatures d'anneau. Comme nous avons pu le voir, celles-ci nécessitent bien souvent de multiples calculs cryptographiques bien souvent rédhibitoires pour une utilisation dans un composant aux capacités restreintes. Dans [CC10], j'ai travaillé avec Iwen Coisel sur l'étude de ces briques dans le cas d'une implémentation sur une carte à puce, sans hypothèse d'inviolabilité, mais en imaginant que la carte est aidée, dans les calculs cryptographiques, par une entité intermédiaire considérée comme (beaucoup) plus puissante.

En pratique, cet intermédiaire est par exemple le lecteur de cartes à puce ou bien le téléphone mobile embarquant la carte SIM. Dans chaque cas, il est bien connu que cette entité est plus puissante, en capacité de calcul et en espace de stockage, que la carte à puce

ou la carte SIM. L'idée est donc que l'entité possédant les éléments secrets se fasse aider par l'entité possédant la puissance de calcul.

Ceci a déjà été étudié dans le cas de RSA [MKI88, BQ95], dans le cas d'un vérifieur [GL05] ou bien dans certains cas particulier de signatures de groupe [MB02] ou de DAA [BCC04]. Dans ces deux derniers cas, les auteurs proposent une construction permettant à une carte à puce de réaliser une signature de groupe de type ACJT en coopération avec un intermédiaire plus puissant. Il s'agit donc d'améliorer l'efficacité de la construction initiale tout en ne perdant rien sur les propriétés de sécurité. Pour autant, aucun de ces articles

1. ne propose une modélisation de ce concept ;
2. ne donne clairement les propriétés de sécurité dans le cas où cet intermédiaire est contrôlé par l'adversaire ;
3. ne montre que la construction donnée est optimale d'un point de vue efficacité.

Nous avons donc pallié ce problème avec Iwen Coisel, lors de sa thèse, en formalisant le concept de cryptographie anonyme assistée. Dans notre étude, nous nous sommes placés dans le cas le plus courant que l'on rencontre dans les architectures actuelles, à savoir que l'intermédiaire va se mettre en coupure entre le prouveur et le vérifieur. En effet, je rappelle que le prouveur est censé avoir peu de ressources et il s'agit donc en pratique d'une carte à puce ou une carte SIM. De l'autre côté, l'intermédiaire va de ce fait correspondre au lecteur de carte où l'ordinateur connecté au lecteur, ou bien le téléphone mobile.

Nous avons étudié en détails le cas des signatures de groupe, des signatures aveugles et des signatures d'anneaux. Ces trois briques possèdent des propriétés de sécurité communes, comme la consistance, la validité et l'intraçabilité et certaines d'entre elles possèdent en plus des propriétés de sécurité spécifiques. Il est nécessaire d'étudier en détail ces propriétés de sécurité dans le cas de la cryptographie assistée puisque, comme nous allons le voir dans les cas concrets, l'intermédiaire peut dans certains cas avoir des connaissances particulières que personne n'a dans le modèle que je qualifierai de "standard".

- La **consistance** permet de s'assurer qu'une entité légitime est acceptée avec une probabilité écrasante. L'intermédiaire pose ici un réel problème puisqu'il se place en coupure entre le prouveur et le vérifieur. Il peut ainsi envoyer des éléments qui vont amener le vérifieur a systématiquement refuser un prouveur légitime. Nous nous sommes donc placés dans le cas où l'intermédiaire est considéré de confiance du point de vue de la consistance[1].
- La **validité** implique qu'une entité non légitime ne soit acceptée par le vérifieur qu'avec une probabilité négligeable. Dans le cas qui nous intéresse, l'adversaire va pouvoir prendre le contrôle du vérifieur et ainsi obtenir des informations qui ne seraient normalement pas disponibles.
- Dans l'expérience d'**intraçabilité**, l'adversaire va être capable d'interagir avec plusieurs prouveurs, va pouvoir en corrompre certains, obtenir les clés secrètes d'autres avant d'en donner deux de son choix en sortie. Il joue alors une authentification avec l'un des deux prouveurs et doit décider lequel des deux a pris part à cette authentification. Dans notre cas, nous avons décidé de distinguer deux cas.
  - **Intraçabilité faible** : de la même façon que pour la consistance, le prouveur va

1. Ce qui ne sera pas nécessairement le cas pour les autres propriétés.

faire confiance à l'intermédiaire. Ainsi, l'intraçabilité ne va pas être vérifiée vis-à-vis de cette entité. Ceci est à nouveau cohérent dans certains cas pratiques, comme celui où le prouveur est une carte SIM et l'intermédiaire est un téléphone mobile. En effet, ce dernier dispose d'autres moyens plus classiques d'obtenir des informations identifiant la carte SIM.
Dans ce cas, nous avons montré qu'un schéma dans notre modèle était intraçable si et seulement si le même schéma sans intermédiaire est intraçable.

– **Intraçabilité forte** : dans ce cas là, l'intermédiaire est corrompu par l'adversaire. Or, l'intermédiaire a pour but d'aider les prouveurs dans leurs calculs et peut donc potentiellement obtenir des informations non disponibles par les autres acteurs du système. Il est clair que ces informations supplémentaires peuvent lui permettre de reconnaître un prouveur qu'il a déjà vu, et donc casser l'intraçabilité.

La "cryptographie coopérative" a pour but de gagner en efficacité par rapport au cas où le prouveur effectue seul l'ensemble des calculs cryptographiques. Nous avons donc défini le **gain calculatoire** d'un protocole incluant l'intermédiaire par rapport au même protocole où le prouveur effectue seul les calculs. Il est clair qu'à partir d'un protocole initial, il est possible de définir plusieurs protocoles assistés différents, chacun proposant une répartition des calculs différente entre le prouveur et l'intermédiaire. Nécessairement, certaines de ces variantes ne sont pas sûres alors que d'autres ne sont pas optimales. Parmi toutes les variantes sûres, il en existe une qui est la meilleure. Nous avons donc défini le meilleur gain calculatoire qui correspond au gain de l'unique variante assistée qui est à la fois optimale et sûre.

Nous avons ensuite étudié plusieurs protocoles existants, proposant des signatures de groupe, des signatures aveugles à anonymat révocable ou des signatures d'anneau, en étudiant une version aidée faible (avec une intraçabilité faible) et forte (avec une intraçabilité forte). Nous avons obtenu des résultats très variés, comme le montre le Tableau 4.1.

Schéma	Version	Complexité
signature de groupe XSGS [DP06]	Standard	11 exponentiations et 1 couplage
	Faible	1 exponentiation
	Forte	11 exponentiations
signature d'anneau CGS [CGS07]	Standard	$\nu^2 + 9\nu + 6$ exponentiations [2]
	Faible	9ν exponentiations
	Forte	$\nu^2 + 9\nu + 6$ exponentiations
signature aveugle HT [HT07]	Standard	55 exponentiations et 3 couplages
	Faible	2 exponentiations
	Forte	10 exponentiations et 2 couplages

FIG. 4.1 – *Bilan sur la cryptographie coopérative*

En résumé,

1. il est possible d'obtenir des versions faibles réellement très intéressantes du point de vue gain calculatoire. C'est le cas du schéma de signature de groupe de Delerablée-Pointcheval [DP06] où on gagne au final un facteur vingt ;

2. $\nu = \sqrt{N}$ où N est la taille de l'anneau.

2. les versions fortes ne donnent souvent qu'un gain minime, même si pour certaines, comme à nouveau la version aidée forte de Delerablée-Pointcheval, on gagne cette fois-ci un facteur trois ;
3. pour certaines versions, fortes ou faibles, il apparaît que les versions aidées ne sont pas intéressantes du tout, comme par exemple les résultats obtenus pour le schéma de signature d'anneau de Chandran-Groth-Sahai [CGS07].

Globalement, l'étude qu'Iwen Coisel et moi-même avons effectuée permet d'avoir un modèle complet pour la cryptographie aidée, prenant en compte les signatures de groupe, les signatures aveugles et les signatures d'anneau. Elle montre aussi qu'il est possible de trouver la meilleure variante assistée à l'aide du gain calculatoire optimal. Nous avons enfin montré que pour certaines constructions, l'approche de la cryptographie assistée n'est pas intéressante en pratique puisque la meilleure variante possible ne donne qu'un gain minime.

4.2 Modèle et Constructions pour les Systèmes RFID

Les systèmes RFID apportent depuis bientôt dix ans des problématiques nouvelles au domaine de la cryptographie. Les étiquettes RFID ont pour objectif de remplacer les codes barres. Elles pourraient aussi être amenées à remplir de nombreux nouveaux services que les codes barres ne peuvent pas faire, comme par exemple assurer un suivi après la vente d'un produit. Pourtant, cela ne doit pas se faire au détriment d'une part de la sécurité et d'autre part de la vie privée du possesseur de l'objet en question. Pour ce dernier point, si on imagine qu'une personne porte continuellement sur elle une étiquette RFID et que celle-ci soit lisible pas n'importe quel lecteur, cette personne peut alors être tracée dans le moindre de ses mouvements, sans même en être consciente. Pour autant, il est nécessaire (sinon, l'étiquette devient inutile et peut donc être détruite) que certains lecteurs autorisés soient capables de lire l'étiquette en question, lorsque cela est nécessaire.

4.2.1 Modélisation de la Protection de la Vie Privée

Marc Girault et moi-même avons demandé à Iwen Coisel d'étudier la modélisation des systèmes RFID avec protection de la vie privée des utilisateurs. Bien que d'autres modèles de ces systèmes fussent déjà existants [Avo05, JW07, LBdM07, Vau07], aucun ne paraissait être suffisamment complet, jusqu'à l'article de Vaudenay [Vau07]. Celui-ci a présenté un modèle très complet, mais aussi très complexe, capable d'inclure la plupart des attaques que pourrait prévoir un adversaire contre ces systèmes.

Ainsi, un adversaire du système va être capable d'interagir avec plusieurs étiquettes différentes en même temps, mais sans pour autant savoir à quelles étiquettes il a affaire. Selon les définitions de Vaudenay, une étiquette libre (après appel à l'oracle Free) n'est pas dans le champ d'action de l'adversaire et ce dernier ne peut donc pas interagir avec elle, alors qu'une étiquette affectée peut être manipulée par l'adversaire. Par ailleurs, une même étiquette libre peut à nouveau être affectée. Son pseudonyme, c'est-à-dire l'identifiant par lequel l'adversaire la connaît, va nécessairement être différent d'une affectation à l'autre, et donc sans lien apparent pour l'adversaire.

Avec Iwen Coisel et Marc Girault [CCG10], nous avons restreint l'adversaire aux attaques

courantes sur la protection de la vie privée. Notre modèle est ainsi moins fort que celui de Vaudenay. Pourtant, il est plus facile à aborder et à utiliser pour prouver la sécurité d'un système.

Notre modèle part du principe qu'un adversaire contre la protection de la vie privée va chercher à établir des liens non triviaux entre les étiquettes. Nous distinguons alors trois cas différents, illustrés par la Figure 4.2 :

- un lien non trivial standard est un lien entre deux étiquettes jamais corrompues affectées à des moments différents ;
- un lien non trivial passé est un lien entre une étiquette corrompue (via l'oracle CORRUPT) et une affectation passée de cette étiquette ;
- un lien non trivial futur est un lien entre une étiquette précédemment corrompue et une affectation courante de cette étiquette.

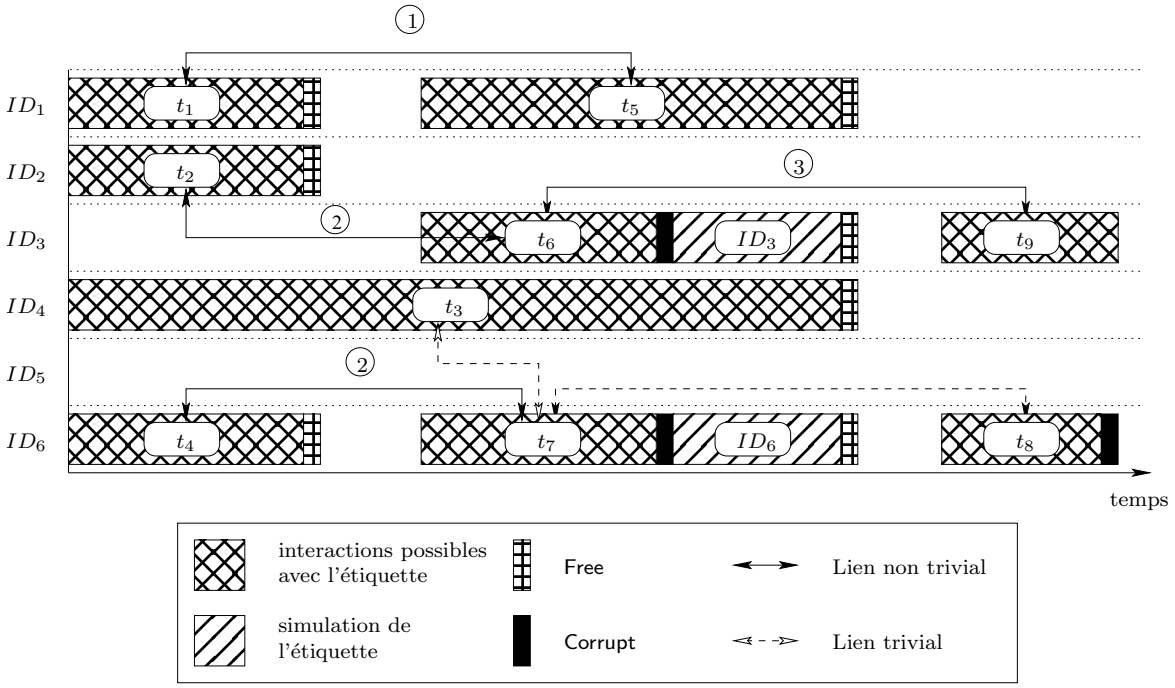

FIG. 4.2 – *Modèle pour la Protection de la Vie Privée*

Nous avons montré que seul un adversaire sans restriction sur les oracles est capable de trouver des liens non triviaux futurs. Si un tel adversaire n'existe pas, nous disons alors que le schéma vérifie la propriété de **protection de la vie privée forte**. Cette notion est très proche de celle définie par Vaudenay puisqu'un adversaire sans restriction définit de même la protection de la vie privée forte, que ce soit dans notre modèle ou dans celui de Vaudenay. Compte tenu des remarques faites ci-dessus, notre notion est moins forte que celle de Vaudenay.

Par ailleurs, Vaudenay a montré que sa propre propriété de protection de la vie privée forte n'est pas atteignable [Vau07]. De notre côté, nous avons montré que notre définition de protection de la vie privée forte était atteignable en exhibant un schéma (voir ci-dessous).

Celle-ci se place donc entre les protections de la vie privée forte et destructive (un appel à l'oracle de corruption d'une étiquette détruit cette dernière) définies par Vaudenay.

4.2.2 Constructions Basées sur la Cryptographie à Clé Secrète

Iwen Coisel et moi-même nous sommes par ailleurs intéressés aux constructions de systèmes RFID, et dans un premier temps, à celles basées sur des infrastructures à clé secrète. Dans cette section, je me place donc dans le cas où chaque étiquette RFID partage avec le lecteur légitime une clé secrète tk. Dans ce contexte, il existe deux types de constructions pour obtenir un bon niveau de protection de la vie privée : le premier consiste à randomiser les échanges alors que le second modifie la clé partagée après chaque authentification.

Le mécanisme de base de ce dernier type est celui proposé par Okhubo *et al.* [OSK05] avec le système OSK. Celui-ci utilise simplement deux fonctions de hachage : la première $\mathcal{H}_1$ permet à l'étiquette RFID de répondre à une requête du lecteur en calculant $\mathcal{H}_1(\mathsf{tk})$ et la seconde, $\mathcal{H}_2$, permet aux deux protagonistes de mettre à jour la clé après l'envoi précédent, par $\mathsf{tk} = \mathcal{H}_2(\mathsf{tk})$. La vérification pour un lecteur légitime consiste à rechercher dans sa base de données la bonne clé, en utilisant à chaque fois la fonction $\mathcal{H}_1$. Une fois cette clé retrouvée, l'étiquette est identifiée et le lecteur met à son tour la clé de cette étiquette à jour dans sa base.

Lors de la conférence RFIDSec [CC08], Iwen Coisel et moi-même avons apporté des éléments à la modélisation des systèmes RFID dans le cas des infrastructures à clé secrète avec mise à jour de clé. En effet, en étudiant le système OSK ci-dessus, nous avons fait deux constats :

- un adversaire est capable de mettre à jour une étiquette légitime suffisamment de fois pour qu'elle soit finalement refusée par le lecteur. En effet, l'étiquette va mettre à jour sa clé à chaque requête et si celle-ci provient d'un adversaire, le lecteur légitime ne va pas le faire. En imaginant que le lecteur teste chaque étiquette de sa base un nombre fixé m de fois (comme préconisé dans [OSK05] ou décrit dans la variante OSK-AO [AO05]), l'adversaire a simplement à interagir avec l'étiquette légitime $m + 1$ fois pour qu'elle soit ensuite systématiquement refusée par le lecteur ;
- un adversaire peut envoyer une valeur totalement aléatoire au lecteur, dans le but de lancer une attaque de type déni de service. En effet, le lecteur va chercher une correspondance avec un étiquette légitime mais ne va pas la trouver (à moins de trouver une collision sur une fonction de hachage, ce qui est considéré comme infaisable). Si la valeur m est trop grande, le travail du lecteur va être très important et peut être rédhibitoire.

Nous avons donc défini le nombre maximum de désynchronisations qu'un adversaire peut engendrer sur un système, ainsi que la capacité de resynchronisation du système, c'est-à-dire le nombre de désynchronisations que le système (côté lecteur et côté étiquette) est capable de récupérer. Un système sera ainsi considéré comme bon si la capacité de resynchronisation du système est supérieure à la capacité de désynchronisation de l'adversaire. Ainsi, compte tenu des remarques ci-dessus, le système OSK n'est pas bon.

Il est aussi possible de comparer les systèmes de ce type par le nombre de tests que doit faire en moyenne le lecteur pour retrouver une étiquette légitime.

Après une étude des systèmes existants selon les critères précédents, nous avons, avec Iwen

Coisel [CC08], proposé un nouveau système, baptisé C^2, ayant de bonnes caractéristiques et basé sur le principe de OSK. Il est représenté dans la Figure 4.3 où R est le lecteur et $\mathcal{T}$ l'étiquette RFID.

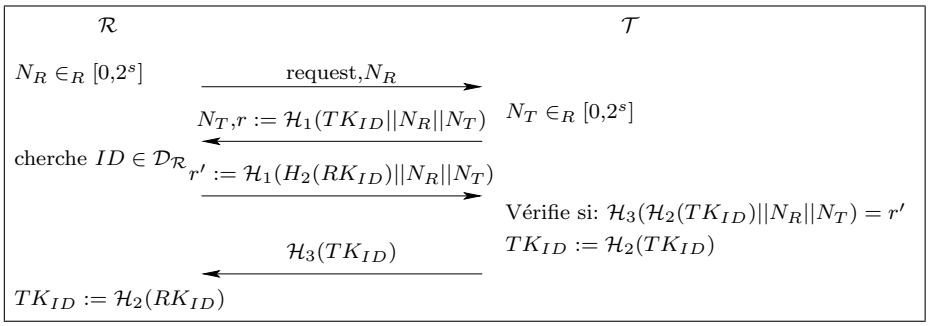

FIG. 4.3 – *Système RFID C^2*

Informellement, notre système propose une solution aux problèmes liés à OSK en permettant à l'étiquette RFID d'authentifier le lecteur avant de mettre à jour sa clé secrète. Le lecteur attend ensuite la confirmation par l'étiquette qu'elle a mis à jour sa clé avant de le faire de son côté. Le nombre maximum de desynchronisations qu'un adversaire peut obtenir est ainsi limité à 1.

Du point de vue de la protection de la vie privée, les schémas basés sur la cryptographie à clé secrète n'atteignent cependant pas le niveau le plus intéressant. En effet, soit les propriétés de resynchronisation ne sont pas optimales, soit ils sont sujets à un attaque permettant de trouver un lien non trivial passé, comme c'est le cas pour le schéma C^2. Plus précisément, si l'adversaire affecte deux étiquettes, écoute les communications d'une authentification d'une des deux étiquettes (sans savoir laquelle), puis corrompt aléatoirement l'une des deux, il est alors capable de faire le lien entre l'authentification et l'étiquette corrompue à l'aide du dernier message $\mathcal{H}_3(\mathsf{tk})$ envoyé par l'étiquette [BBEG09].

Il semble difficile de mettre au point un système RFID avec une bonne protection de la vie privée et basé sur une infrastructure à clé secrète. Lors de la thèse d'Iwen Coisel, nous avons alors étudié le cas des infrastructures à clé publique.

4.2.3 Constructions Basées sur la Cryptographie à Clé Publique

La cryptographie à clé publique n'est a priori pas adaptée au monde des RFID puisqu'elle utilise des techniques mathématiques beaucoup plus lourdes que celles de la cryptographie à clé secrète. Pourtant, Marc Girault et David Lefranc [GL04] ont été les premiers à montrer que c'était possible pour le système d'authentification GPS [GPS06].

Une première idée pourrait consister à utiliser les techniques des signatures de groupe pour obtenir les propriétés désirées. Cependant, les versions standards ne sont clairement pas implémentables dans une étiquette RFID. Il faut alors se retourner vers les solutions que j'ai présentées dans la Section 4.1. J'ai ainsi proposé avec Benoît Calmels, Marc Girault et Hervé Sibert [CCGS06] un tel système. Il décrit comment une étiquette peut être détectée, s'authentifier ou être identifiée, selon les connaissances du lecteur qu'elle a en face d'elle. Le

problème de cette solution est qu'elle nécessite une étiquette ayant la propriété d'inviolabilité, ce qui est une hypothèse très forte en pratique.

De son côté, Vaudenay a proposé dans [Vau07] un système RFID avec protection de la vie privée basé sur l'utilisation d'un schéma de chiffrement à clé publique. Dans ce système, le lecteur possède la clé secrète de déchiffrement et chaque étiquette légitime est capable de chiffrer des messages pour le lecteur, à l'aide de l'algorithme Enc. De façon très simple, à la réception d'une requête accompagnée d'un aléa a, l'étiquette va chiffrer son identifiant unique tid à l'aide de la clé publique de chiffrement : $\textsc{Enc}(a\|\mathsf{tid})$. Le lecteur va ainsi être capable de déchiffrer le message reçu et de vérifier la valeur de l'identifiant. Le protocole complet est décrit dans la Figure 4.4.

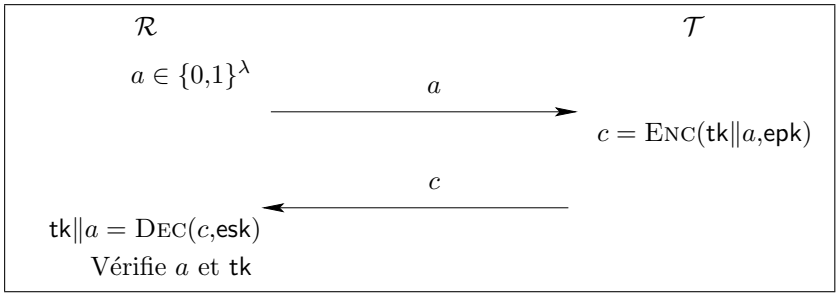

Fig. 4.4 – *Protocole de Vaudenay*

Vaudenay a par ailleurs prouvé que son système était sûr, sous l'hypothèse que le schéma de chiffrement est IND-CCA. Dans son mémoire de thèse, Iwen Coisel a par ailleurs prouvé que ce même schéma générique ne permettait pas à un adversaire de trouver des liens triviaux passés, à nouveau si le schéma de chiffrement utilisé est IND-CCA.

De façon très informelle, la validité du schéma est obtenue en utilisant le fait qu'un adversaire n'est pas capable, à partir d'un chiffré valide (en l'occurrence le chiffré d'un identifiant valide et d'un aléa), de générer le chiffré d'un autre message relié au précédent (et donc de répondre à une nouvelle authentification). En effet, comme le schéma de chiffrement est IND-CCA et que l'adveraire a tout le temps accès à l'oracle de déchiffrement, il est aussi non-malléable (NM-CCA). La protection de la vie privée est prouvée en utilisant l'indistinguabilité (IND-CCA), et donc l'impossibilité pour l'adversaire de distinguer parmi deux messages (et donc deux identifiants d'étiquettes) lequel est effectivement chiffré lors d'une authentification.

Avec Iwen Coisel et Jonathan Etrog [CCE10], nous avons étudié des constructions concrètes de ce système générique. Dans un premier temps, nous avons étudié l'implémentation de ce système à l'aide du schéma de chiffrement DHAES [ABR98].

Ensuite, nous avons remarqué que la propriété IND-CCA était trop forte. Dans le schéma ci-dessus, cette propriété est vraisemblablement trop forte puisque l'adversaire ne peut malléer aucun type de message, le schéma de chiffrement étant NM-CCA. Dans le cas qui nous intéresse, il est suffisant que le schéma de chiffrement vérifie la propriété que nous avons appelé dans [CCE10] la "non-malléabilité constante et fixée". Celle-ci décrit le fait qu'un adversaire peut avoir accès à la clé publique de chiffrement et à un oracle qui prend en entrée une valeur a et donne en sortie le chiffré de $a\|\mathsf{tid}$ où tid est un élément secret. L'objectif de l'adversaire consiste alors à sortir le chiffré de $\tilde{a}\|\mathsf{tid}$ où $\tilde{a}$ lui est imposé. Nous avons montré

que cette non-malléabilité était suffisante pour prouver la validité du schéma, selon le même principe que celui donné ci-dessus.

Pour certains schémas de chiffrement qui ne sont clairement pas IND-CCA, tels que Rabin ou El Gamal, nous n'avons pas été en mesure de savoir s'ils vérifiaient ou non cette propriété. Pour autant, le schéma de chiffrement de Rabin a par exemple été utilisé pour mettre en place le schéma d'authentification WIPR [OF08], après certaines modifications dues à Shamir [Sha08]. A l'opposé, pour le schéma de chiffrement El Gamal haché, il est possible d'exhiber une attaque contre son utilisation dans le schéma d'authentification RFID de Vaudenay.

Pour finir, j'ai proposé avec Iwen Coisel et Marc Girault [CCG10] un schéma basé sur le schéma de chiffrement El Gamal haché (par exemple [CMPP06]) et permettant d'obtenir la preuve qu'un adversaire n'est pas capable de donner un lien non trivial futur. Pour cela, nous avons utilisé un MAC, noté f, comme décrit dans la Figure 4.5. Dans ce protocole, $\mathsf{sk}[\mathsf{ID}]$ et $\mathsf{k}[\mathsf{ID}]$ sont des clés propres à l'étiquette avec l'identifiant ID et $\mathcal{H}$ une fonction de hachage.

Par contre, nous ne sommes pas parvenus à savoir si le schéma de chiffrement sous-jacent obtenu est finalement IND-CCA ou non. Notre utilisation du "MAC then Sign" tombe en effet dans la catégorie pour laquelle on ne sait pas conclure.

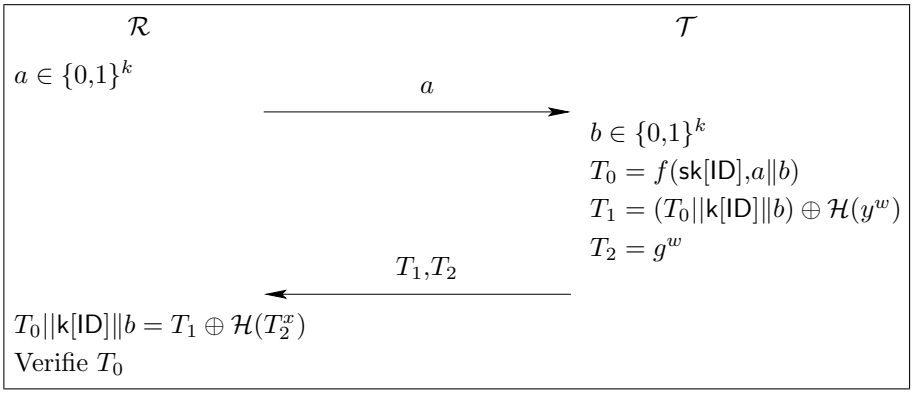

FIG. 4.5 – *Le protocole Canard-Coisel-Girault*

Nous avons enfin fait un comparatif des schémas existants, obtenu en étudiant d'une part la taille de l'implémentation en termes de portes équivalentes (GE pour *Gate Equivalent*) et d'autre part l'efficacité de l'authentification au moment de son exécution. Pour les schémas que nous avons proposés, nous préconisons d'utiliser la technique des coupons, proposée par le projet européen CAFE (ESPRIT 7023) et utilisée notamment dans [GL04]. Celle-ci remplace le calcul d'une exponentiation modulaire par le stockage du résultat pour une utilisation ultérieure. Ceci entraîne un gain non négligeable en efficacité. Nos résultats sont répertoriés dans la Table 4.6. Pour notre implémentation, nous utilisons notamment, pour les besoins du MAC, l'algorithme à clé secrète PRESENT [BKL+07, RPLP08].

	Taille implémentation [GE]	Temps [ms]	Protection vie privée	Utilisation coupons
Batina *et al.* [BGK+07]	< 7000	< 600	Non	Non
GPS [GL04, MR07]	1541	28	Non	Oui
WIPR [OF08]	5706	≈ 600	Non	Non
WIPR-SAEP [OF08, WS09]	8035	?	Forte	Non
Notre schéma (avec coupons)	< 2500	< 100	Forte	Oui
Notre schéma (sans coupon)	≈ 18000	≈ 300	Forte	Non

FIG. 4.6 – *Bilan sur l'éfficacité*

4.3 Conclusion et Perspectives

L'hypothèse d'inviolabilité d'une carte à puce est très puissante. Elle permet d'obtenir des schémas largement plus efficaces que les schémas standards et donne ainsi des systèmes pratiques et implémentables dès aujourd'hui. Bien qu'acceptable dans certains cas, elle a pourtant parfois ses limites : si par exemple une carte est cassée, alors bien souvent le système complet tombe en même temps. Il serait toutefois intéressant d'étudier la réalisation de schémas où l'hypothèse d'inviolabilité ne porte que sur une seule propriété de sécurité. Par exemple, dans le cas de la monnaie électronique, il est peut-être possible de mettre en place un système pour lequel si une carte à puce est cassée, alors seule la propriété d'anonymat tombe, mais pas celle de non falsification des pièces.

La cryptographie assistée permet de se passer de cette hypothèse d'inviolabilité. L'étude que j'ai menée avec Iwen Coisel permet de se faire une idée très précise des modifications à apporter à un schéma existant afin d'obtenir la meilleure variante sûre possible. Pourtant, je pense que notre étude mériterait d'être complétée de résultats plus précis sur le calcul des gains calculatoires. Il serait par exemple intéressant de prendre en compte des valeurs réelles côté carte à puce (ou carte SIM) et côté intermédiaire (PC ou téléphone mobile), mais aussi la perte éventuellement occasionnée par les échanges entre le prouveur et l'intermédiaire, parfois plus importants dans le cas assisté.

Dans le monde des RFID, le travail à fournir est encore important. La modélisation qu'Iwen Coisel a apportée couvre de nombreuses attaques et semble suffisamment simple pour être utilisée en pratique. En ce qui concerne les constructions, les schémas actuels ne sont cependant pas tous satisfaisants. En effet, ceux qui sont basés sur la cryptographie à clé secrète n'obtiennent pas un bon niveau de protection de la vie privée. Ceux basés sur la cryptographie à clé publique, comme celui que nous avons proposé avec Iwen Coisel et Marc Girault, et qui est à ce jour, et à ma connaissance, le plus efficace, utilise la technique très controversée des coupons. En effet, cela entraîne obligatoirement des difficultés, ou au moins des adaptations, au niveau du rechargement de ces coupons, voire des problèmes de sécurité. Il est donc important de continuer la recherche dans ce domaine, soit en trouvant des alternatives aux coupons, soit en trouvant une façon astucieuse de calculer et recharger ces derniers.

Conclusion

La recherche en cryptographie est en continuelle évolution et il devient de plus en plus difficile de maîtriser tous les domaines qu'elle couvre. Depuis ma thèse entre 2000 et 2003, j'ai effectué ma recherche dans un thème qui me permet de toucher à un maximum de briques cryptographiques : la protection de la vie privée.

Mes travaux portent sur des services existants ou à venir, dans des marchés plus ou moins matures, dans le but de les rendre plus sûrs, plus rapides et plus simples à utiliser. Il reste beaucoup de points à résoudre mais c'est un domaine très porteur. En effet, un manque de respect des règles de protection de la vie privée des individus pourrait entraîner une réaction de rejet de beaucoup d'innovations. Par ailleurs, de plus en plus de textes législatifs voient le jour, obligeant les fournisseurs de services à fournir des solutions pour protéger les données personnelles de leurs clients.

Mon activité de chercheur porte donc à la fois sur les services que je suis amené à étudier ou à développer dans le cadre de mon travail, mais aussi sur les briques cryptographiques que ces services doivent utiliser. Mon but est de rendre ces services sûrs et de faire en sorte que la vie la vie privée de leurs utilisateurs soit protégée.

En conclusion, la cryptographie est un outil extrêmement puissant capable de répondre à de nombreux besoins, dont ceux liés à cette "privacy". La recherche doit continuer dans ce domaine et l'objectif que je me donne pour les prochaines années est de construire de nouveaux services ou d'améliorer ceux existants en utilisant la cryptographie moderne.

Bibliographie

[ABR98] Michel Abdalla, Mihir Bellare, and Philip Rogaway. DHAES: an encryption scheme based on the Diffie-Hellman problem. Technical Report, UC Davis Computer Science, 1998.

[ACdMT05] Giuseppe Ateniese, Daniel H. Chou, Breno de Medeiros, and Gene Tsudik. Sanitizable signatures. In *ESORICS 2005*, volume 3679 of *Lecture Notes in Computer Science*, pages 159–177. Springer, 2005.

[ACJM10] Pascal Avondes, Sébastien Canard, Amandine Jambert, and Michel Milhau. New model and architecture for private networks. In submission, 2010.

[ACJT00] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 255–270. Springer, 2000.

[AF96] Masayuki Abe and Eiichiro Fujisaki. How to date blind signatures. In *ASIACRYPT'96*, volume 1163 of *Lecture Notes in Computer Science*, pages 244–251. Springer, 1996.

[AO05] Gildas Avoine and Philippe Oechslin. A scalable and provably secure hash-based RFID protocol. In *PerCom Workshops 2005*, pages 110–114. IEEE Computer Society, 2005.

[ASM07] Man Ho Au, Willy Susilo, and Yi Mu. Practical compact e-cash. In *ACISP 2007*, volume 4586 of *Lecture Notes in Computer Science*, pages 431–445. Springer, 2007.

[ASM08] Man Ho Au, Willy Susilo, and Yi Mu. Practical anonymous divisible e-cash from bounded accumulators. In *Financial Cryptography 2008*, volume 5143 of *Lecture Notes in Computer Science*, pages 287–301. Springer, 2008.

[Avo05] Gildas Avoine. Adversarial model for radio frequency identification. Cryptology ePrint Archive, Report 2005/049, 2005. `http://eprint.iacr.org/`.

[BB08] Dan Boneh and Xavier Boyen. Short signatures without random oracles and the sdh assumption in bilinear groups. *J. Cryptology*, 21(2):149–177, 2008.

[BBEG09] Côme Berbain, Olivier Billet, Jonathan Etrog, and Henri Gilbert. An efficient forward private RFID protocol. In *ACM Conference on Computer and Communications Security 2009*. ACM, 2009.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2004.

[BCC04] Ernest F. Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In *ACM Conference on Computer and Communications Security 2004*, pages 132–145. ACM, 2004.

[BCG05] François Boudet, Sébastien Canard, and Stéphane Guilloteau. Procédé d'anonymisation des données personnelles dans le domaine de la santé. Brevet n^o 05 00784, 2005.

[BCP07] Emmanuel Bresson, Olivier Chevassut, and David Pointcheval. Provably secure authenticated group Diffie-Hellman key exchange. *ACM Transactions on Information and System Security*, 10(3), 2007.

[BFF+09] Christina Brzuska, Marc Fischlin, Tobias Freudenreich, Anja Lehmann, Marcus Page, Jakob Schelbert, Dominique Schröder, and Florian Volk. Security of sanitizable signatures revisited. In *Public Key Cryptography 2009*, volume 5443 of *Lecture Notes in Computer Science*, pages 317–336. Springer, 2009.

[BGK+07] Lejla Batina, Jorge Guajardo, Tim Kerins, Nele Mentens, Pim Tuyls, and Ingrid Verbauwhede. Public-key cryptography for RFID-tags. In *PerCom Workshops 2007*, pages 217–222. IEEE Computer Society, 2007.

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF formulas on ciphertexts. In *TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 325–341. Springer, 2005.

[BKL+07] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and Charlotte Vikkelsoe. PRESENT: an ultra-lightweight block cipher. In *CHES 2007*, volume 4727 of *Lecture Notes in Computer Science*, pages 450–466. Springer, 2007.

[Bou00] Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In *EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 431–444. Springer, 2000.

[BQ95] Philippe Béguin and Jean-Jacques Quisquater. Fast server-aided RSA signatures secure against active attacks. In *CRYPTO'95*, volume 963 of *Lecture Notes in Computer Science*, pages 57–69. Springer, 1995.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security 1993*, pages 62–73, 1993.

[BSZ05] Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of group signatures: the case of dynamic groups. In *CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 136–153. Springer, 2005.

[CC08] Sébastien Canard and Iwen Coisel. Data synchronization in privacy-preserving RFID authentication schemes. In *RFID Sec 2008*, pages 88–99. Manfred Aigner editor, 2008.

[CC10] Sébastien Canard and Iwen Coisel. Server aided cryptography for anonymity. In submission, 2010.

[CCE10] Sébastien Canard, Iwen Coisel, and Jonathan Etrog. Lighten encryption schemes for secure and private RFID systems. In *Workshop on Lightweight Cryptography 2010*. To appear, 2010.

[CCG10] Sébastien Canard, Iwen Coisel, and Marc Girault. Privacy-preserving RFID identification systems: model and construction. In submission, 2010.

[CCGS06] Benoît Calmels, Sébastien Canard, Marc Girault, and Hervé Sibert. Low-cost cryptography for privacy in RFID systems. In *CARDIS 2006*, volume 3928 of *Lecture Notes in Computer Science*, pages 237–251. Springer, 2006.

[CCM06] Sébastien Canard, Fabrice Clerc, and Benjamin Morin. A secure universal loyalty card. In *WOSIS 2006*, pages 13–22. INSTICC Press, 2006.

[CCS08] Jan Camenisch, Rafik Chaabouni, and Abhi Shelat. Efficient protocols for set membership and range proofs. In *ASIACRYPT 2008*, volume 5350 of *Lecture Notes in Computer Science*, pages 234–252. Springer, 2008.

[CCT07] Sébastien Canard, Iwen Coisel, and Jacques Traoré. Complex zero-knowledge proofs of knowledge are easy to use. In *ProvSec 2007*, volume 4784 of *Lecture Notes in Computer Science*, pages 122–137. Springer, 2007.

[CD07] Sébastien Canard and Céline Dulong. Preuve de connaissance d'un secret dans un intervalle basée sur la décomposition binaire. Brevet n^o 07 07002, 2007.

[CDG+09] Sébastien Canard, Cécile Delerablée, Aline Gouget, Emeline Hufschmitt, Fabien Laguillaumie, Hervé Sibert, Jacques Traoré, and Damien Vergnaud. Fair e-cash: be compact, spend faster. In *ISC 2009*, volume 5735 of *Lecture Notes in Computer Science*, pages 294–309. Springer, 2009.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO'94*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer, 1994.

[CG02] Sébastien Canard and Marc Girault. Implementing group signature schemes with smart cards. In *CARDIS 2002*, pages 1–10. USENIX, 2002.

[CG07] Sébastien Canard and Aline Gouget. Divisible e-cash systems can be truly anonymous. In *EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 482–497. Springer, 2007.

[CG08] Sébastien Canard and Aline Gouget. Anonymity in transferable e-cash. In *ACNS 2008*, volume 5037 of *Lecture Notes in Computer Science*, pages 207–223. Springer, 2008.

[CG10] Sébastien Canard and Aline Gouget. Multiple denominations in e-cash with compact transaction data. In *Financial Cryptography and Data Security 2010*, Lecture Notes in Computer Science. Springer, 2010.

[CGH06] Sébastien Canard, Aline Gouget, and Emeline Hufschmitt. A handy multi-coupon system. In *ACNS 2006*, volume 3989 of *Lecture Notes in Computer Science*, pages 66–81, 2006.

[CGH07] Sébastien Canard, Aline Gouget, and Emeline Hufschmitt. Handy compact e-cash system. In *SAR SSI 2007*, pages 319–332, 2007.

[CGS07] Nishanth Chandran, Jens Groth, and Amit Sahai. Ring signatures of sub-linear size without random oracles. In *ICALP 2007*, volume 4596 of *Lecture Notes in Computer Science*, pages 423–434. Springer, 2007.

[CGT04] Sébastien Canard, Matthieu Gaud, and Jacques Traoré. A new fair blind signature process. Brevet n^o 04 290 558.8, 2004.

[CGT06] Sébastien Canard, Matthieu Gaud, and Jacques Traoré. Defeating malicious servers in a blind signatures based voting system. In *Financial Cryptography 2006*,

volume 4107 of *Lecture Notes in Computer Science*, pages 148–153. Springer, 2006.

[CGT08] Sébastien Canard, Aline Gouget, and Jacques Traoré. Improvement of efficiency in (unconditional) anonymous transferable e-cash. In *Financial Cryptography 2008*, volume 5143 of *Lecture Notes in Computer Science*, pages 202–214. Springer, 2008.

[Cha82] David Chaum. Blind signatures for untraceable payments. In *CRYPTO'82*, pages 199–203. Plenum Press, 1982.

[CHL05] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact e-cash. In *EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 302–321. Springer, 2005.

[CJ08] Sébastien Canard and Amandine Jambert. Group key management: from a non-hierarchical to a hierarchical structure. In *INDOCRYPT 2008*, volume 5365 of *Lecture Notes in Computer Science*, pages 213–225. Springer, 2008.

[CJ09] Sébastien Canard and Amandine Jambert. On extended sanitizable signature schemes. In submission, 2009.

[CJ10a] Sébastien Canard and Amandine Jambert. How to protect customers' privacy in billing systems. In submission, 2010.

[CJ10b] Sébastien Canard and Amandine Jambert. Untraceability and profiling in multi-service subscription systems. In submission, 2010.

[CKPM05] Scott Cantor, John Kemp, Rob Philpott, and Eve Maler. Assertions and protocol for the OASIS security assertion markup language (SAML). OASIS Standard, 2005.

[CKS09] Jan Camenisch, Markulf Kohlweiss, and Claudio Soriente. An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In *Public Key Cryptography 2009*, volume 5443 of *Lecture Notes in Computer Science*, pages 481–500. Springer, 2009.

[CKY09] Jan Camenisch, Aggelos Kiayias, and Moti Yung. On the portability of generalized Schnorr proofs. In *EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 425–442. Springer, 2009.

[CL02a] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 61–76. Springer, 2002.

[CL02b] Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In *SCN 2002*, volume 2576 of *Lecture Notes in Computer Science*, pages 268–289. Springer, 2002.

[CL04] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 56–72. Springer, 2004.

[CLM08] Sébastien Canard, Fabien Laguillaumie, and Michel Milhau. Trapdoor sanitizable signatures and their application to content protection. In *ACNS 2008*, volume 5037 of *Lecture Notes in Computer Science*, pages 258–276. Springer, 2008.

[CMPP06] Benoît Chevallier-Mames, Pascal Paillier, and David Pointcheval. Encoding-free El Gamal encryption without random oracles. In *Public Key Cryptography 2006*, volume 3958 of *Lecture Notes in Computer Science*, pages 91–104. Springer, 2006.

[CMT08] Sébastien Canard, Eric Malville, and Jacques Traoré. Identity federation and privacy: one step beyond. In *Digital Identity Management 2008*, pages 25–32. ACM, 2008.

[CMT09] Sébastien Canard, Eric Malville, and Jacques Traoré. A client-side approach for privacy-preserving identity federation. *Identity in the Information Society*, 2009.

[Coi09] Iwen Coisel. *Authentification et Anonymat à Bas-Coût : modélisations et Protocoles*. PhD thesis, Université de Caen, 2009.

[CP92] David Chaum and Torben P. Pedersen. Transferred cash grows in size. In *EUROCRYPT'92*, volume 658 of *Lecture Notes in Computer Science*, pages 390–407. Springer, 1992.

[CS06a] Sébastien Canard and Hervé Sibert. How to fit cryptographic e-voting into smart cards. In *FEE 2006*, pages 59–69. IAVOSS, 2006.

[CS06b] Sébastien Canard and Hervé Sibert. Votinbox, a voting system based on smart cards. In *Workshop on e-voting in the UK 2006*, pages 5–12. e-Science Institute, 2006.

[CSST06] Sébastien Canard, Berry Schoenmakers, Martijn Stam, and Jacques Traoré. List signature schemes. *Discrete Applied Mathematics*, 154(2):189–201, 2006.

[CT03a] Sébastien Canard and Jacques Traoré. List signature schemes and application to electronic voting. In *WCC 2003*, pages 81–90, 2003.

[CT03b] Sébastien Canard and Jacques Traoré. On fair e-cash systems based on group signature schemes. In *ACISP 2003*, volume 2727 of *Lecture Notes in Computer Science*, pages 237–248. Springer, 2003.

[CT04] Sébastien Canard and Jacques Traoré. Anonymous services using smart cards and cryptography. In *CARDIS 2004*, pages 83–98. Kluwer, 2004.

[DF02] Ivan Damgård and Eiichiro Fujisaki. A statistically-hiding integer commitment scheme based on groups with hidden order. In *ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 125–142. Springer, 2002.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. In *IEEE Transactions on Information Theory*, volume IT-22(6), pages 644–654. IEEE, 1976.

[DP06] Cécile Delerablée and David Pointcheval. Dynamic fully anonymous short group signatures. In *VIETCRYPT 2006*, volume 4341 of *Lecture Notes in Computer Science*, pages 193–210. Springer, 2006.

[DY05] Yevgeniy Dodis and Aleksandr Yampolskiy. A verifiable random function with short proofs and keys. In *Public Key Cryptography 2005*, volume 3386 of *Lecture Notes in Computer Science*, pages 416–431, 2005.

[Fia97] Amos Fiat. Batch RSA. *J. Cryptology*, 10(2):75–88, 1997.

[FO97] Eiichiro Fujisaki and Tatsuaki Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *CRYPTO'97*, volume 1294 of *Lecture Notes in Computer Science*, pages 16–30. Springer, 1997.

[FPV09] Georg Fuchsbauer, David Pointcheval, and Damien Vergnaud. Transferable anonymous constant-size fair e-cash. In *CANS 2009*, volume To appear of *Lecture Notes in Computer Science*. Springer, 2009.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: practical solutions to identification and signature problems. In *CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.

[Fuc09] Georg Fuchsbauer. Automorphic signatures in bilinear groups. Cryptology ePrint Archive, Report 2009/320, 2009. `http://eprint.iacr.org/`.

[Gam84] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18. Springer, 1984.

[GL04] Marc Girault and David Lefranc. Public key authentication with one (online) single addition. In *CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 413–427. Springer, 2004.

[GL05] Marc Girault and David Lefranc. Server-aided verification: theory and practice. In *ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 605–623. Springer, 2005.

[GPS06] Marc Girault, Guillaume Poupard, and Jacques Stern. On the fly authentication and signature schemes based on groups of unknown order. *J. Cryptology*, 19(4):463–487, 2006.

[Gro05] Jens Groth. Non-interactive zero-knowledge arguments for voting. In *ACNS 2005*, volume 3531 of *Lecture Notes in Computer Science*, pages 467–482. Springer, 2005.

[Gro07] Jens Groth. Fully anonymous group signatures without random oracles. In *ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 164–180. Springer, 2007.

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 415–432. Springer, 2008.

[HT07] Emeline Hufschmitt and Jacques Traoré. Fair blind signatures revisited. In *Pairing 2007*, volume 4575 of *Lecture Notes in Computer Science*, pages 268–292. Springer, 2007.

[JW07] Ari Juels and Stephen A. Weis. Defining strong privacy for RFID. In *PerCom Workshops 2007*, pages 342–347. IEEE Computer Society, 2007.

[KKLA01] Kwangjo Kim, Jinho Kim, Byoungcheon Lee, and Gookwhan Ahn. Experimental design of worldwide internet voting system using pki. In *SSGRR 2001*, 2001.

[KL06] Marek Klonowski and Anna Lauks. Extended sanitizable signatures. In *ICISC 2006*, volume 4296 of *Lecture Notes in Computer Science*, pages 343–355. Springer, 2006.

[KPT04] Yongdae Kim, Adrian Perrig, and Gene Tsudik. Tree-based group key agreement. *ACM Transactions on Information and System Security*, 7(1):60–96, 2004.

[KTY04] Aggelos Kiayias, Yiannis Tsiounis, and Moti Yung. Traceable signatures. In *EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 571–589. Springer, 2004.

[LBdM07] Tri Van Le, Mike Burmester, and Breno de Medeiros. Universally composable and forward-secure RFID authentication and authenticated key exchange. In *ASIACCS 2007*, pages 242–252. ACM, 2007.

[Lib] Alliance Liberty. `http://www.projectliberty.org/`.

[Lip03] Helger Lipmaa. On diophantine complexity and statistical zero-knowledge arguments. In *ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 398–415. Springer, 2003.

[LKKR03] Sangwon Lee, Yongdae Kim, Kwangjo Kim, and DaeHyun Ryu. An efficient tree-based group key agreement using bilinear map. In *ACNS 2003*, volume 2846 of *Lecture Notes in Computer Science*, pages 357–371. Springer, 2003.

[MB02] Greg Maitland and Colin Boyd. Co-operatively formed group signatures. In *CT-RSA 2002*, volume 2271 of *Lecture Notes in Computer Science*, pages 218–235. Springer, 2002.

[MKI88] Tsutomu Matsumoto, Koki Kato, and Hideki Imai. Speeding up secret computations with insecure auxiliary devices. In *CRYPTO'88*, volume 403 of *Lecture Notes in Computer Science*, pages 497–506. Springer, 1988.

[MR07] Máire McLoone and Matthew J. B. Robshaw. Public key cryptography and RFID tags. In *CT-RSA 2007*, volume 4377 of *Lecture Notes in Computer Science*, pages 372–384. Springer, 2007.

[Ngu05] Lan Nguyen. Accumulators from bilinear pairings and applications. In *CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 275–292. Springer, 2005.

[NS00] Toru Nakanishi and Yuji Sugiyama. Unlinkable divisible electronic cash. In *ISW 2000*, volume 1975 of *Lecture Notes in Computer Science*, pages 121–134. Springer, 2000.

[OF08] Yossef Oren and Martin Feldhofer. WIPR - public key identification on two grains and sans. In *RFID Sec 2008*. Manfred Aigner editor, 2008.

[OSK05] Miyako Ohkubo, Koutarou Suzuki, and Shingo Kinoshita. RFID privacy issues and technical challenges. *Commun. ACM*, 48(9):66–71, 2005.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 1999.

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *J. Cryptology*, 13(3):361–396, 2000.

[RPLP08] Carsten Rolfes, Axel Poschmann, Gregor Leander, and Christof Paar. Ultra-lightweight implementations for smart devices - security for 1000 gate equivalents. In *CARDIS 2008*, volume 5189 of *Lecture Notes in Computer Science*, pages 89–103. Springer, 2008.

[RST01] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In *ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 552–565. Springer, 2001.

[Sch89] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In *CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252. Springer, 1989.

[SCPY94] Alfredo De Santis, Giovanni Di Crescenzo, Giuseppe Persiano, and Moti Yung. On monotone formula closure of SZK. In *FOCS'94*, pages 454–465. IEEE, 1994.

[Sha08] Adi Shamir. Squash - a new MAC with provable security properties for highly constrained devices such as RFID tags. In *FSE 2008*, volume 5086 of *Lecture Notes in Computer Science*, pages 144–157. Springer, 2008.

[SPC95] Markus Stadler, Jean-Marc Piveteau, and Jan Camenisch. Fair blind signatures. In *EUROCRYPT'95*, volume 921 of *Lecture Notes in Computer Science*, pages 209–219. Springer, 1995.

[Sta96] Markus Stadler. Publicly verifiable secret sharing. In *EUROCRYPT'96*, volume 1070 of *Lecture Notes in Computer Science*, pages 190–199. Springer, 1996.

[STW00] Michael Steiner, Gene Tsudik, and Michael Waidner. Key agreement in dynamic peer groups. In *IEEE Transactions on Parallel and Distributed Systems*, volume 11(8), pages 769–780. IEEE, 2000.

[SY90] Alfredo De Santis and Moti Yung. Crptograpic applications of the non-interactive metaproof and many-prover systems. In *CRYPTO'90*, volume 537 of *Lecture Notes in Computer Science*, pages 366–377. Springer, 1990.

[Tra99] Jacques Traoré. Group signatures and their relevance to privacy-protecting off-line electronic cash systems. In *ACISP'99*, volume 1587 of *Lecture Notes in Computer Science*, pages 228–243. Springer, 1999.

[vA90] Hans van Antwerpen. *Electronic Cash.* PhD thesis, CWI, 1990.

[Vau07] Serge Vaudenay. On privacy models for RFID. In *ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 68–87. Springer, 2007.

[WS09] Jiang Wu and Doug Stinson. How to improve security and reduce hardware demands of the WIPR RFID protocol. In *IEEE RFID 2009*. IEEE Computer Society, 2009.

[XY04] Shouhuai Xu and Moti Yung. Accountable ring signatures: a smart card approach. In *CARDIS 2004*, pages 271–286. Kluwer, 2004.

Annexe A

Curriculum Vitæ

A.1 Etat Civil

Sébastien CANARD, né le 30 mai 1976 à Lyon, marié, père de 3 enfants (Louis, né le 30 janvier 2004, Eléonore, née le 29 novembre 2006 et Gautier, né le 12 septembre 2009) et demeurant à Caen.

A.2 Formation Initiale et Complémentaire

– **Doctorat d'informatique de l'Université de Caen Basse-Normandie (2003)**

- Sujet : Signatures de groupe, variantes et applications
- Directeur de thèse : Marc Girault (France Télécom R&D)
- Jury : Jacques Stern (ENS, président), David Pointcheval (ENS, rapporteur), Jacques Patarin (Université de Versailles Saint Quentin en Yvelines, rapporteur), Jean-Jacques Quisquater (UCL), Marc Joye (Gemplus)

– **Diplôme d'ingénieur ENSICAEN, option Informatique (2000)**

– **DEA Intelligence Artificielle et Algorithmique de l'Université de Caen (2000) :** option Algorithmique et Protection de l'Information

– **Classes préparatoires aux grandes écoles MP (1994-1997) :** lycée La Martinière Monplaisir à Lyon

– **Baccalauréat C (1994) :** lycée Colbert à Lyon, mention assez bien

A.3 Expérience Professionnelle

– **Ingénieur de Recherche - Orange Labs R&D à Caen (2003 à aujourd'hui)**

  – **Missions** : développer l'expertise du Groupe France Télécom en **cryptographie** afin d'assurer la **sécurité des services**, en particulier les services de **protection des données personnelles**. Analyser les algorithmes, protocoles, services, et logiciels de sécurité existants.

  – **Réalisations** : **publication de 21 articles et dépôt de 18 brevets** à ce jour. Ces innovations ont pour but de fournir de nouveaux services sécurisés au Groupe France Télécom, mais aussi de montrer l'excellence internationale du Groupe dans le domaine de la sécurité des services. Présentation des résultats dans des conférences internationales ou lors de séminaires internes ou universitaires.

    Expertises effectuées, provenant de demandes émanant d'autres entités du Groupe France Télécom. Ces expertises correspondent à des demandes ponctuelles nécessitant des réponses à échéances courtes (environ 1 semaine) ou plus longues (environ 1 mois). Elles **portent essentiellement sur des besoins cryptographiques** (validation d'un outil, aide au choix d'un algorithme, formation, etc.).

    Montage et gestion du projet ANR PACE qui vise à concevoir et fournir des outils permettant la sécurisation des services liés aux transactions électroniques, tout en protégeant au mieux la vie privée des acteurs du système et sans pour autant augmenter les risques de fraude. Il doit ainsi mettre en avant de nouvelles problématiques, trouver les solutions cryptographiques nécessaires et chercher des constructions efficaces au niveau algorithmique.

    Responsable de lots dans **divers projets collaboratifs avec des financements européens** (projet SPICE sur la mise en place d'*enabler* pour les téléphones mobiles) ou de l'ANR et RNRT (projets CRYPTO++ sur les briques cryptographiques pour l'anonymat et SAVE sur le vote électronique).

    Rédaction de **livres blancs** permettant aux décideurs de Orange Labs de connaître un domaine spécifique, le marché possible et la concurrence afin de prendre une décision sur les stratégies à adopter.

    Rédaction de **spécifications** dans le but de développer des solutions techniques. La suite donnée consiste la plupart du temps à encadrer des stages et des **prestations** pour l'**implémentation** de briques cryptographiques ou de démonstrateurs (vote électronique, VoD, gestion de l'identité).

    Encadrement de deux thèses, l'une portant sur les technologies pour la protection de la **vie privée** dans les **systèmes bas-coût** (cartes à puce, SIM) et très bas-coût (**étiquettes RFID**) et l'autre traitant des protocoles cryptographiques pour la sécurité des groupes.

  – **Compétences** : **cryptographie**, sécurité des services, **protection de la vie privée**, protection des données, authentification, **anonymat**, **gestion de l'identité**, paiement, vote électronique.

  – Collaborations : Cryptolog International, ENS, Gemalto, LIX, NXP, Oberthur C.S., Orange Business Services - IT&Labs, Orange Business Services - Solutions santé (anciennement Almerys), STMicroelectronics, ST-Ericsson, Supélec, Université de Caen Basse-Normandie.

– **Doctorant en cryptographie - France Télécom R&D à Caen (2000-2003)**

  – **Missions** : recherche appliquée sur les protocoles cryptographiques de communication sécurisée de groupes.
  – **Réalisations** : 4 publications, dépôt de 5 brevets et présentation des résultats dans des conférences internationales. Montage d'un projet interne au Groupe France Télécom sur la protection de la vie privée dans les services web.
  – **Compétences** : cryptographie, signature électronique avec anonymat, cartes à puce, enchères, monnaie électronique, vote électronique.

A.4 Langues Etrangères

– **Anglais :** courant (TOEIC : 835/1000), pratique quotidienne de la lecture et de l'écriture, 2 à 3 voyages par an pour assister à et/ou faire des présentations dans des conférences internationales.

– **Espagnol :** notions de base, capable de comprendre et de se faire comprendre dans les pays hispaniques.

A.5 Activités d'Encadrement et de Recherche

– **Activités d'enseignement**

  – **Cours de cryptographie au Mastère ENSICAEN - IHESI** à Caen : 18 heures par an de 2003 à 2006. Initiation à la cryptographie pour des étudiants avec des profils d'ingénieur ou littéraires, le but étant qu'ils soient au final familiers avec les outils cryptographiques et leurs utilisations.
  – **Cours de cryptographie à clé secrète en Licence Pro** "MCA" - IUT Mesures Physiques à Caen : 6 heures par an depuis 2003. Approche technique de la cryptographie à clé secrète avec apprentissage des propriétés désirées, des algorithmes usuels et de leurs utilisations en pratique.
  – **Formation d'une journée de l'équipe de développeurs** de la société Almerys afin qu'ils comprennent les bases de la cryptographie en vue des développements d'une nouvelle application au sein de leur société basée sur des outils cryptographiques.

– **Responsabilités de recherche**

  – **Responsable du projet collaboratif PACE** : ce projet fait partie de l'appel à **projets Télécommunications de l'ANR** et il a par ailleurs été labellisé par le **pôle de compétitivité bas normand TES**. Il a débuté fin 2008 pour une durée de 4 ans et un effort total de **300 hommes mois** sur l'ensemble du projet. Il réunit des **partenaires académiques** (Ecole Normale Supérieure, CNRS, Université de Caen Basse-Normandie) et **industriels** (France Télécom, Gemalto, ST-Ericsson, Cryptolog) et traite de la cryptographie pour la monnaie électronique et applications bilinéaires.

- **Responsabilités dans les projets :** je suis ou j'ai été responsable de lots dans divers **projets collaboratifs nationaux** (financement ANR ou RNRT pour les projets CRYPTO++ et SAVE) ou **européens** (financement **FP6 IST** pour le projet SPICE) avec la fourniture des livrables du lot et la gestion du travail des intervenants du projet.

– **Activités d'encadrement de stages**

J'encadre chaque année un ou deux stagiaires étudiant soit en Master (professionnel ou de recherche), soit en école d'ingénieur. Ces stages me permettent de déléguer certaines tâches et d'approfondir des points importants de ma recherche. Ils me donnent aussi l'occasion de partager mes méthodes de travail avec de futurs ingénieurs ou chercheurs, entre autres la communication, la recherche bibliographique ou la rédaction.

- **Messagerie instantanée et échange de clé (année 2004) :** encadrement de deux stages (en master et ingénieur) ayant pour but de mettre en place un système de messagerie instantanée sécurisé à l'aide d'un système cryptographique d'échange de clé généralisé à plusieurs personnes. Ces stages ont abouti à la mise en place d'**un démonstrateur**.
- **Mots de passe jetables en contexte de mobilité (année 2004) :** encadrement d'un stage ingénieur portant sur la technique des mots de passe jetables pour la mise en place d'un service de paiement de parcmètre via un téléphone mobile. Ce stage a abouti à **un brevet** et à la mise en place d'**un démonstrateur**.
- **Développement d'un système de vote électronique (année 2005) :** encadrement de deux stages (master et ingénieur) pour la création d'un nouveau système de vote électronique où chaque électeur possède une carte à puce qui a été développée par la société STMicroelectronics. Ce stage a permis de développer ce **nouveau système de vote** qui a été présenté lors du salon international CARTES.
- **Mise en place et étude de la sécurité d'un schéma de signature électronique avec anonymat (année 2005) :** encadrement d'un stage en master de recherche sur la création de nouvelles briques cryptographiques pour l'anonymat.
- **Etude de l'optimisation d'une implémentation Java d'un schéma de signature de groupe (année 2006) :** encadrement d'un stage en master de développement visant à optimiser, en terme d'efficacité, des implémentations Java existantes de briques cryptographiques.
- **Preuve de sécurité d'un mécanisme générique de production d'une preuve de connaissance (année 2006) :** encadrement d'un stage en master de recherche qui a permis de prouver la sécurité d'une construction générique d'une preuve de connaissance à divulgation nulle de connaissance.
- **Gestion des contenus numériques protégés dans un contexte familial (année 2007) :** encadrement d'un stage en master de recherche dont le but est d'étudier une solution de protection des contenus achetés ou autoproduits au sein d'une famille avec contrôle parental et gestion des amis. Ce stage a abouti à la mise en place d'**un démonstrateur** et au dépôt d'**un brevet**.
- **Preuve de connaissance d'un secret pouvant avoir un nombre fini de valeurs (année 2007) :** encadrement d'un stage en master de recherche visant à améliorer les preuves de connaissance d'un nombre entier secret appartenant à un intervalle. Ce stage a abouti au dépôt d'**un brevet**.

- **Mise en conformité de l'implémentation de briques cryptographiques (année 2008) :** encadrement d'un stage ingénieur ayant abouti à l'homogénéisation des implémentations cryptographiques existantes et à la création d'un document de référence inter-entreprise (règles de nommage, commentaires et organisation des classes) pour l'implémentation de briques cryptographiques en Java.
- **Multi-exponentiation et application à la monnaie électronique divisible (année 2008) :** encadrement d'un stage en master de recherche sur l'amélioration d'un système de monnaie électronique divisible au niveau de la génération des clés du système. Ce stage a abouti à diverses améliorations de l'existant et la rédaction d'un **article est en cours**.

– **Activités d'encadrement de thèses**

Depuis trois ans, je co-encadre trois thèses à hauteur de 70%, la direction de ces thèses étant effectuée par Marc Girault (affilié au laboratoire GREYC à l'Université de Caen Basse-Normandie) et Gilles Zémor (laboratoire IMB à l'Université de Bordeaux), et David Pointcheval (Ecole Normale Supérieure). Mon travail consiste à participer aux choix des sujets de recherche, des soumissions dans les journaux ou les conférences et à permettre aux doctorants de formaliser leurs travaux.

- **Mécanismes de protection de la vie privée destinés à des dispositifs à ressources limitées (2006 à 2009) :** cette thèse, préparée par Iwen Coisel, traite des différentes propriétés de sécurité pouvant être atteintes dans un protocole d'**authentification** (et/ou d'identification) d'un dispositif à ressources limitées (de type **carte à puce** ou **étiquette RFID**) et plus particulièrement aux propriétés assurant la **protection de la vie privée** des possesseurs du dispositif.

  Cette thèse a été soutenue le 9 octobre 2009. Elle a abouti à **trois publications** dans des conférences internationales, à deux articles supplémentaires en cours de **soumission**, ainsi qu'au dépôt de **deux brevets**.
- **Cryptographie pour la protection des données personnelles (démarrée en 2007) :** cette thèse, préparée par Amandine Jambert, a pour but d'étudier les briques cryptographiques dans le contexte de la **protection des données personnelles** des utilisateurs. Il s'agit d'étudier, d'améliorer et de créer les briques possédant des propriétés particulières telles que la délégation de signature, ou la **gestion de clés** dans un groupe. Le doctorant étudie par la suite les applications de ces briques cryptographiques dans le cadre de la protection de la vie privée

  Cette thèse a pour l'instant abouti à **une publication** dans des conférences internationales, à trois soumissions en cours, ainsi qu'au dépôt de **deux brevets**.
- **Authentification, anonymat et révocation (démarrée en 2009) :** cette thèse, préparée par Roch Lescuyer, traite des attestations anonymes d'attributs d'utilisateurs (nom, adresse, âge, étudiant, handicapé, etc.) et du principe du *need-to-know*. Il s'agit d'étudier les mécanismes cryptographiques de base utilisés dans ces systèmes et notamment la possibilité de révocation d'un utilisateur.

A.6 Publications et Brevets

Ma recherche est principalement axée sur la cryptographie et la sécurité des services. Je m'intéresse tout particulièrement aux briques cryptographiques complexes telles que les signatures de groupe ou les signatures aveugles mais aussi des techniques plus standards telles que les preuves de connaissance ou la gestion de clés dans les groupes.

Une fois ces briques mises en place, je les utilise pour créer ou renforcer des services en protégeant la vie privée des utilisateurs, le plus souvent (mais pas exclusivement) en leur permettant d'être anonyme. Je propose ainsi de nouveaux systèmes de vote électronique, de gestion d'identité ou de monnaie électronique. Je possède notamment de nombreux résultats majeurs dans ce dernier domaine avec des articles dans les plus grandes conférences internationales en cryptographie et la levée de plusieurs verrous majeurs tels qu'un schéma de monnaie électronique divisible réellement anonyme ou la création du premier schéma de monnaie électronique transférable efficace.

J'ai aussi obtenu des résultats dans l'étude des dispositifs à ressources limitées tel qu'une carte à puce ou une étiquette RFID, notamment avec Iwen Coisel qui a récemment soutenu sa thèse sous ma direction. Dans ces cas, les capacités de calcul et de stockage étant restreints, il est nécessaire de trouver des solutions adaptées, tout en essayant d'obtenir le même niveau de sécurité.

– Publications dans des Revues

- Sébastien Canard, Berry Schoenmakers, Martijn Stam, and Jacques Traoré. List signature schemes. Discrete Applied Mathematics, 154(2):189-201, 2006.
- Sébastien Canard, Eric Malville, and Jacques Traoré. A client-side approach for privacy-preserving identity federation. Identity in the Information Society, 2009.

– Publications dans des Conférences

- Sébastien Canard and Aline Gouget. Multiple denominations in e-cash with compact transaction data. In Financial Cryptography and Data Security 2010, volume to appear of Lecture Notes in Computer Science. Springer, 2010.
- Sébastien Canard, Cécile Delerablée, Aline Gouget, Emeline Hufschmitt, Fabien Laguillaumie, Hervé Sibert, Jacques Traoré, and Damien Vergnaud. Fair ecash: be compact, spend faster. In ISC 2009, volume 5735 of Lecture Notes in Computer Science, pages 294-309. Springer, 2009.
- Sébastien Canard and Amandine Jambert. Group key management: from a nonhierarchical to a hierarchical structure. In INDOCRYPT 2008, volume 5365 of Lecture Notes in Computer Science, pages 213-225. Springer, 2008.
- Sébastien Canard and Aline Gouget. Anonymity in transferable e-cash. In ACNS 2008, volume 5037 of Lecture Notes in Computer Science, pages 207-223. Springer, 2008.
- Sébastien Canard, Fabien Laguillaumie, and Michel Milhau. Trapdoor sanitizable signatures and their application to content protection. In ACNS 2008, volume 5037 of Lecture Notes in Computer Science, pages 258-276. Springer, 2008.

- Sébastien Canard, Aline Gouget, and Jacques Traoré. Improvement of efficiency in (unconditional) anonymous transferable e-cash. In Financial Cryptography 2008, volume 5143 of Lecture Notes in Computer Science, pages 202-214. Springer, 2008.
- Sébastien Canard, Iwen Coisel, and Jacques Traoré. Complex zero-knowledge proofs of knowledge are easy to use. In ProvSec 2007, volume 4784 of Lecture Notes in Computer Science, pages 122-137. Springer, 2007.
- Sébastien Canard, Aline Gouget, and Emeline Hufschmitt. Handy compact ecash system. In SAR SSI 2007, pages 319-332, 2007.
- Sébastien Canard and Aline Gouget. Divisible e-cash systems can be truly anonymous. In EUROCRYPT 2007, volume 4515 of Lecture Notes in Computer Science, pages 482-497. Springer, 2007.
- Benoît Calmels, Sébastien Canard, Marc Girault, and Hervé Sibert. Low-cost cryptography for privacy in RFID systems. In CARDIS 2006, volume 3928 of Lecture Notes in Computer Science, pages 237-251. Springer, 2006.
- Sébastien Canard, Aline Gouget, and Emeline Hufschmitt. A handy multicoupon system. In ACNS 2006, volume 3989 of Lecture Notes in Computer Science, pages 66-81, 2006.
- Sébastien Canard, Matthieu Gaud, and Jacques Traoré. Defeating malicious servers in a blind signatures based voting system. In Financial Cryptography 2006, volume 4107 of Lecture Notes in Computer Science, pages 148-153. Springer, 2006.
- Sébastien Canard and Jacques Traoré. Anonymous services using smart cards and cryptography. In Jean-Jacques Quisquater, Pierre Paradinas, Yves Deswarte, and Anas Abou El Kalam, editors, CARDIS, pages 83-98. Kluwer, 2004.
- Sébastien Canard and Jacques Traoré. On fair e-cash systems based on group signature schemes. In ACISP 2003, volume 2727 of Lecture Notes in Computer Science, pages 237-248. Springer, 2003.
- Sébastien Canard and Marc Girault. Implementing group signature schemes with smart cards. In CARDIS 2002, pages 1-10. USENIX, 2002.

– Publications dans des Workshops

- Sébastien Canard, Iwen Coisel, and Jonathan Etrog. Lighten encryption schemes for secure and private RFID systems. In Workshop on Lightweight Cryptography 2010. To appear, 2010.
- Sébastien Canard, Eric Malville, and Jacques Traoré. Identity federation and privacy: one step beyond. In Digital Identity Management 2008, pages 25-32. ACM, 2008.
- Sébastien Canard and Iwen Coisel. Data synchronization in privacy-preserving RFID authentication schemes. In RFID Sec 2008, pages 88-99. Manfred Aigner editor, 2008.
- Sébastien Canard and Hervé Sibert. How to fit cryptographic e-voting into smart cards. In FEE 2006, pages 59-69. IAVOSS, 2006.
- Sébastien Canard and Hervé Sibert. Votinbox, a voting system based on smart cards. In Workshop on e-voting in the UK 2006, pages 5-12. e-Science Institute, 2006.
- Sébastien Canard, Fabrice Clerc, and Benjamin Morin. A secure universal loyalty card. In WOSIS 2006, pages 13-22. INSTICC Press, 2006.

- Sébastien Canard, Stéphane Guilloteau, Eric Malville, Jacques Traoré. e-Mask: a practical approach to ensure anonymity and accountability on internet. Chuan-Kun Wu editor, CANS 2003. 2003.
- Sébastien Canard and Jacques Traoré. List signature schemes and application to electronic voting. In WCC 2003, pages 81-90, 2003.

– Soumission en cours

- Sébastien Canard, and Iwen Coisel. Server-aided cryptography for anonymity.
- Sébastien Canard, and Amandine Jambert. On extended sanitizable signature schemes.
- Sébastien Canard, and Amandine Jambert. How to protect customer's privacy in billing systems.
- Sébastien Canard, and Amandine Jambert. Untraceability and profiling in multi-service subscription systems.
- Sébastien Canard, Iwen Coisel, and Marc Girault. Privacy-Preserving RFID Identification Systems: Model and Construction.
- Pascal Avondes, Sébastien Canard, Amandine Jambert, and Michel Milhau. New Model and Architecture for Private Networks.

– Brevets déposés

- S. Canard, A. Jambert. Procédé cryptographique d'abonnement anonyme à un service. Brevet déposé.
- S. Canard, A. Jambert, E. Malville. Procédé cryptographique d'authentification anonyme et d'identification séparée d'un utilisateur. Brevet n°9 53953 déposé le 12/06/2009.
- S. Canard, I. Coisel, M. Girault. Procédé d'authentification ultra-rapide avec protection de la vie privée. Brevet n° 08 53956 déposé le 16/06/2008.
- S. Canard, I. Coisel. Procédé d'authentification d'une étiquette radio par un lecteur radio. Brevet n° 08 53055 déposé le 09/05/2008.
- S. Canard, C. Dulong. Preuve de connaissance d'un secret dans un intervalle basée sur la décomposition binaire. Brevet n° 07 07002 déposé le 05/10/2007.
- S. Canard, A. Jambert, M. Milhau. Procédé de gestion de clés cryptographiques dans des graphes dynamiques. Brevet déposé le 20/09/2007.
- S. Canard, M. Milhau. Procédé de délégation de signature. Brevet n° 07 03650 déposé le 23/05/2007.
- S. Canard, S. Guilloteau, E. Malville, J. Traoré. Procédé d'authentification unique d'un utilisateur auprès de fournisseurs de service. Brevet n°07 53556 déposé le 28/02/2007.
- S.Canard, C. Delerablée, H. Sibert. Procédé de signature anonyme traçable et compact. Brevet n° 06 55983 déposé le 29/12/2006.
- S.Canard, C. Delerablée, H. Sibert. Procédé de signature anonyme traçable sans levée d'anonymat. Brevet n° 06 55987 déposé le 29/12/2006.
- S.Canard, F. Clerc, B. Morin. Sécurisation de données pour programmes de fidélisation de clientèle. Brevet n° 05 02144 déposé le 03/03/2005.
- F. Boudet, S.Canard, S. Guilloteau. Procédé d'anonymisation des données personnelles dans le domaine de la santé. Brevet n° 05 00784 déposé le 26/01/2005.

- B. Calmels, S. Canard, M. Girault, H. Sibert. Procédé de contrôle de niveaux de confidentialité pour RFID. Brevet n° 04 12681 déposé le 30/11/2004.
- B. Calmels, S. Canard, D. Picquenot, A. Saif, H. Sibert. Système de paiement par suite de jetons. Brevet n° 04 06604 déposé le 17/06/2004.
- S. Canard, M. Girault, J. Traoré. Procédé et système de signature de liste. Brevet n° 04 7212 déposé le 19/05/2004.
- S. Canard, S. Caron, B. Cosnefroy, E. Malville, J. Traoré. Procédé de paiement anonyme et sécurisé sur internet et mobile. Brevet n° 04 05402 déposé le 18/05/2004.
- S. Canard, M. Gaud, J. Traoré. Electronic voting process using fair blind signatures. Brevet n° 04 290 557.0 déposé le 02/03/2004.
- S. Canard, M. Gaud, J. Traoré. A new fair blind signature process. Brevet n° 04 290 558.8 déposé le 02/03/2004.
- S Canard, S. Guilloteau, E. Malville, J. Traoré. Procédé d'accès à un service avec authentification rapide et anonymat révocable et systèmes de maintien de session. Brevet n° 02 14230 déposé le 14/11/2002.
- S. Canard, M. Girault, J. Traoré. Procédé de signature de liste et application au vote électronique. Brevet n° 02 09218 déposé le 19/07/2002.
- S. Canard, M. Girault, J. Traoré. Procédé cryptographique de révocation à l'aide d'une carte à puce. Brevet n° 02 00569 déposé le 17/01/2002.
- S. Canard, M. Girault, J. Traoré. Procédé et dispositif de signature anonyme au moyen d'une clé privée partagée. Brevet n° 02 00107 déposé le 04/01/2002.
- D. Arditti, S. Canard, M. Girault, J. Traoré. Système cryptographique de signature de groupe. Brevet n° 01 16950 déposé le 27/12/2001.

Annexe B

Résumés de mes Articles

B.1 Contributions aux Outils Cryptographiques

Complex zero-knowledge proofs of knowledge are easy to use (ProvSec 2007)

Since 1985 and their introduction by Goldwasser, Micali and Rackoff, followed in 1988 by Feige, Fiat and Shamir, zero-knowledge proofs of knowledge have become a central tool in modern cryptography. Many articles use them as building blocks to construct more complex protocols, for which security is often hard to prove. The aim of this paper is to simplify analysis of many of these protocols, by providing the cryptographers with a theorem which will save them from stating explicit security proofs. Kiayias, Tsiounis and Yung made a first step in this direction at Eurocrypt'04, but they only addressed the case of so-called "triangular set of discrete-log relations". By generalizing their result to any set of discrete-log relations, we greatly extend the range of protocols it can be applied to.

List signature schemes and application to electronic voting (WCC 2003)

An electronic voting scheme is a mechanism in which voters can securely vote and several authorities can collect these votes to calculate the result. Several voting schemes exist and most of them are based on three cryptographic concepts: mix-nets, blind signatures or homomorphic encryption. Another potential primitive for electronic voting is given by group signature schemes, which allow a group member to anonymously sign messages on behalf of the group. Unfortunately, such signature schemes are not directly usable in electronic voting. This is why we introduce in this paper a variant of group signature schemes, called (electoral) list signature schemes, in which the signatures cannot be opened by anybody and are linkable if (and only if) they have been produced within the same "sequence" (typically an election day). We give a concrete example of a list signature scheme. Then, we design an electronic voting scheme based on this new concept. Our solution is surprisingly simple (only one connection to a database) and efficient (only a few proofs of knowledge), while satisfying the fundamental needs of security in electronic voting. Finally, it is a very convenient solution for both on-line voting (via Internet for example) and off-line voting (using electronic voting booths).

List signature schemes (Journal DAM)

A group signature scheme allows members of a group to issue signatures on behalf of the group, while hiding for each signature which group member actually issued it. A group signature scheme also involves a group manager, who is able to open any group signature by showing which group member issued it.

We introduce the concept of list signatures as a variant of group signatures which set a limit on the number of signatures each group member may issue. These limits must be enforced *without* having the group manager open signatures of honest group members—which excludes the trivial solution in which the group manager opens every signature to see whether some group members exceed their limits. Furthermore, we consider the problem of *publicly identifying* group members who exceed their limits, also *without* involving the group manager.

Trapdoor sanitizable signatures and their application to content protection (ACNS 2008)

Sanitizable signatures allow a designated entity to modify some specific parts of a signed message and to produce a new signature of the resulting message without any interaction with the original signer. In this paper, we extend these sanitizable signatures to formally introduce *trapdoor sanitizable signatures*. In this concept, the power of sanitization is given to possibly several entities, for a given message/signature by using a trapdoor computed by the signer at any time. We also give a generic construction of such trapdoor sanitizable signatures. Eventually, we apply our new cryptographic tool to group content protection, permitting members of the group to distribute a protected content among themselves.

On extended sanitizable signature schemes (en soumission)

Sanitizable signature schemes allow a semi-trusted entity to modify some specific portions of a signed message while keeping a valid signature of the original signer and without interacting with the signer. In this paper, we give a new secure sanitizable signature scheme which is, to the best of our knowledge, the most efficient construction with a such high level of security. We also enhance the Brzuska *et al.* model on sanitizable signature schemes by adding new features. We thus model the way to limit the set of possible modifications on a single block, the way to force the same modifications on different admissible blocks, and the way to limit both the number of modifications of admissible blocks and the number of versions of a signed message. We finally present two cryptanalysis on proposals for two of these features due to Klonowski and Lauks at ICISC 2006 and propose some practical constructions for two of them.

Group key management: from a non-hierarchical to a hierarchical structure (Indocrypt 2008)

Since the very beginnings of cryptography many centuries ago, key management has been one of the main challenges in cryptographic research. In case of a group of players wanting to share a common key, many schemes exist in the literature, managing groups where all players

are equal or proposing solutions where the group is structured as a hierarchy. This paper presents the first key management scheme suitable for a hierarchy where no central authority is needed and permitting to manage a graph representing the hierarchical group with possibly several roots. This is achieved by using a HMAC and a non-hierarchical group key agreement scheme in an intricate manner and introducing the notion of virtual node.

B.2 Protection de la Vie Privée dans les Services

e-Mask: a practical approach to ensure anonymity and accountability on internet (CANS 2003)

The mechanism most of today's web sites (e.g. on-line auctions, on-line games) provide to ensure the anonymity of the users is based on the use of a pseudonym and a password. This approach comes with a number of limitations. The probably most critical of them is that it does not provide accountability, a property that becomes essential as high valuable financial transactions are at stake.

This paper introduces a protocol, called e-Mask, which provides both anonymity and accountability in an efficient way. Our protocol mainly relies on a combination of an anonymous authentication mechanism (based on either anonymous certificates, fair blind signatures or group signatures) and an authentication session management mechanism (based on one-time passwords). This article describes the e-Mask protocol itself.

Defeating malicious servers in a blind signatures based voting system (Financial Cryptography 2006)

In this paper, we present two failures in the blind signatures based voting system Votopia which has been tested during the last World Soccer Cup. We then propose a fix which relies on *fair blind signatures*. The resulting scheme is practical, satisfies the fundamental needs of security in electronic voting, including *public verifiability*, and compares favorably with other like systems in terms of computational cost. As an illustration, our variant of Votopia has been successfully trialed during the French referendum on the European Constitution in May 2005.

A client-side approach for privacy-preserving identity federation (ACM-DIM 2008 et Journal IIS)

Providing Single Sign-On (SSO) between service providers and enabling service providers to share user personal attributes are critical for both users to benefit from a seamless access to their services, and service providers to realize new business opportunities. Today, however, the users have several independent, partial identities spread over different service providers. Providing SSO and attribute sharing requires that links (federations) are established between (partial) identities. In SAML 2.0, the links between identities are stored and managed at the network side by the identity providers (network-side identity federation). This model prevents the service providers from mass-correlating the partial identities they have, but the users must fully trust the identity providers. In this paper, we propose a complementary approach where

the users have a full control of the links between their partial identities. It is a client-side identity federation approach, which relies on the introduction of a new cryptographic tool, called invariable partially blind signature scheme, that may be of independent interest.

New model and architecture for private networks (en soumission)

This paper describes a security architecture based on hierarchical groups allowing flexible management of data access inside a private network. The idea is to define groups and subgroups inside a private network of users in order to facilitate a customized rights management on acquired or private data. Graph models are chosen to set well-defined cryptographic keys used to protect rights objects inside each subgroup. With this approach, a manager of a private network, for example the father in a family, will be able to limit or forbid access rights on acquired protected contents, for the subgroup of children. This work, well-adapted to the OMA DRM domain approach, aims to resolve transfer and exchange requirements inside a small group of users, with respect to given usage rights, and adapted for both connected or non-connected mode. This is the first complete architecture that suits many use cases and private networks such as associations, families, enterprises or academic department.

A secure universal loyalty card (WOSIS 2006)

In this paper, we propose a generic loyalty system based on smart cards which may be implemented in existing devices like cell phones or PDAs. Our loyalty system is secure and offers some desirable features both to customers and vendors, and may further the adoption of such win-win marketing operations. In particular, the system, reliable for both parties, is universal in the sense that there is a one-to-many relationship between a customer's loyalty card and the vendors.

How to protect customers' privacy in billing systems (en soumission)

In this paper, we study the privacy of customers in billing systems. In such systems, customers can use a service provided by some provider of several services but need to pay the bill afterwards. We focus on the privacy property in such systems, that is the infeasibility for any actor of the system to know which customer is paying for which service. More precisely, we give a new approach based on the use of a new cryptographic building block we introduce: sanitizable group signature scheme. Such scheme, of independent interest, permits group members to anonymously sign a message. Moreover, one designated entity is able to open one given signature to retrieve the identity of the signer. Finally, another authorized entity can modify some designated parts of the initial message in such a way that the opening of the new signature is due to the initial group member.

Untraceability and profiling in multi-service subscription systems (en soumission)

In this paper, we introduce the concept of privacy-preserving multi-service subscription systems. With such system, service providers can propose to their customers, by the way

of a subscription, several distinct services that users can access anonymously. We moreover study how users can be anonymous and unlinkable w.r.t. the service provider during the subscription process. Furthermore, our system permits service providers to make profiling on the users's customs, while providing several levels of privacy protection. We propose several instantiations, based on Camenisch-Lysyanskaya signature schemes and need, for this purpose, to introduce one additional feature to these schemes, which permits the user to update one previously obtained signature. This additional result may be of independent interest.

B.3 Monnaie Electronique

On fair e-cash systems based on group signature schemes (ACISP 2003)

A fair electronic cash system is a system that allows customers to make payments anonymously. Moreover, under certain circumstances, a trusted authority can revoke the anonymity of suspicious transactions. Various fair e-cash systems using group signature schemes have been proposed. Unfortunately, they do not realize coin tracing (the possibility to trace the coins withdrawn by a customer). In this paper, we describe several failures in on of these solutions and we present a secure and efficient fair e-cash system based on a group signature scheme. Our system ensures traceability of *double-spenders*, supports coin tracing and provides coins that are unforgeable and anonymous under standard assumptions.

A handy multi-coupon system (ACNS 2006)

A coupon is an electronic data that represents the right to access a service provided by a service provider (e.g. gift certificates or movie tickets). Recently, a privacy-protecting multi-coupon system that allows a user to withdraw a predefined number of single coupons from the service provider has been proposed by Chen et al. at *Financial Crypto 2005*. In this system, every coupon has the same value which is predetermined by the system. The main drawbacks of Chen et al. proposal are that the redemption protocol of their system is inefficient, and that no formal security model is proposed. In this paper, we consequently propose a formal security model for coupon systems and design a practical multi-coupon system with new features: the quantity of single coupons in a multi-coupon is not defined by the system and the value of each coupon is chosen in a predefined set of values.

Handy compact e-cash system (SAR-SSI 2007)

This paper presents an off-line anonymous e-cash scheme with features useful for a practical use. The quantity of coins withdrawn by the user is no longer predetermined by the system whereas the anonymity of users is preserved. Moreover, the number of coins withdrawn by the user is not necessarily a power of two which allows more flexibility. During the withdrawal protocol, the user can from now on choose the value of each coin in a predetermined set of values. This feature is particularly interesting from a practical point of view since it allows to improve the efficiency of both the spending and the deposit protocols. Another result of this paper is the addition of a validity date for each coin in order to reduce the growing of the bank's database after each transaction. The proposed handy compact e-cash scheme

integrates these new features without significatively impact the compactness of the electronic wallet.

Fair e-cash: be compact, spend faster (ISC 2009)

We present the first *fair e-cash system* with a compact wallet that enables users to spend efficiently k coins while only sending to the merchant $\mathcal{O}(\lambda \log k)$ bits, where λ is a security parameter. The best previously known schemes require to transmit data of size at least linear in the number of spent coins. This result is achieved thanks to a new way to use the Batch RSA technique and a tree-based representation of the wallet. Moreover, we give a variant of our scheme with a less compact wallet but where the computational complexity of the spend operation does not depend on the number of spent coins, instead of being linear at best in existing systems.

Divisible e-cash systems can be truly anonymous (Eurocrypt 2007)

This paper presents an off-line divisible e-cash scheme where a user can withdraw a divisible coin of monetary value 2^L that he can parceled and spend anonymously and unlinkably. We present the construction of a security tag that allows to protect the anonymity of honest users and to revoke anonymity only in case of cheat for protocols based on a binary tree structure without using a trusted third party. This is the first divisible e-cash scheme that provides both full unlinkability and anonymity without requiring a trusted third party.

Multiple denominations in e-cash with compact transaction data (Financial Cryptography 2010)

We present a new construction of *divisible* e-cash that makes use of 1) a new generation method of the binary tree of keys; 2) a new way of using bounded accumulators. The transaction data sent to the merchant has a constant number of bits while spending a monetary value 2^ℓ. Moreover, the spending protocol does not require complex zero-knowledge proofs of knowledge such as proofs about double discrete logarithms. We then propose the first strongly anonymous scheme with standard unforgeability requirement and realistic generation parameters while improving the efficiency of the spending phase.

Improvement of efficiency in (unconditional) anonymous transferable e-cash (Financial Cryptography 2008)

The practical advantage expected from transferable e-cash compare to non-transferable is the significant reduction of the interaction number between the bank and the users. However, this property is not fulfilled by *anonymous* transferable e-cash schemes of the state-of-the art. In this paper, we first present a transferable e-cash scheme with a reduced number of communications between the bank and the users that fulfils the *computational anonymity* property. Next, we present a transferable e-cash scheme with a reduced interaction number that fulfils the *unconditional anonymity*. This latter scheme is quite less efficient.

Anonymity in transferable e-cash (ACNS 2008)

Regular cash systems provide both the *anonymity* of users and the *transferability* of coins. In this paper, we study the anonymity properties of transferable e-cash. We define two natural additional levels of anonymity directly related to transferability and not reached by existing schemes that we call *full anonymity (FA)* and *perfect anonymity (PA)*. We show that the FA property can be reached by providing a generic construction and that the PA's cannot. Next, we define two restricted perfect anonymity properties and we prove that it is possible to design a transferable e-cash scheme where a bounded adversary not playing the bank cannot recognize a coin he has already owned.

B.4 La Cryptographie (Ultra) Legère

Implementing group signature schemes with smart cards (CARDIS 2002)

Group signature schemes allow a group member to sign messages on behalf of the group. Such signatures must be anonymous and unlinkable but, whenever needed, a designated group manager can reveal the identity of the signer. During the last decade group signatures have been playing an important role in cryptographic research; many solutions have been proposed and some of them are quite efficient, with constant size of signatures and keys. However, some problems still remain among which the large number of computations during the signature protocol and the difficulty to achieve coalition-resistance and to deal with member revocation. In this paper we investigate the use of a tamper-resistant device (typically a smart card) to efficiently solve those problems.

Anonymous services using smart cards and cryptography (CARDIS 2004)

More and more services provided by Internet pose a problem of privacy and anonymity. One cryptographic tool that could be used for solving this problem is the group signature. Each member of the group is able to anonymously produce a signature on behalf of the group and a designated authority can, in some cases, revoke this anonymity.

During the last decade, many anonymous services using this concept have been proposed: electronic auctions, electronic cash systems, anonymous credentials. But for some other services where the anonymity is essential (such as electronic voting or call for tenders), group signature schemes cannot be applied as they are. For this reason, Canard and Traoré proposed a variant that is partially linkable and not openable, called list signature scheme.

In this paper, we first improve the cryptographic tool of Canard and Traoré by proposing some optional modifications of list signature schemes such as anonymity revocation. We then propose more efficient list signature schemes, by using a smart card to produce the signature. We finally propose some concrete implementations of our proposals. As a result, we obtain more efficient solutions that are useful in many more services.

How to fit cryptographic e-voting into smart cards (FEE 2006)

The complexity of voting procedures make it challenging to design a secure electronic voting system. In many proposals, the security of the system relies mainly on a black box voting machine. Meanwhile, the most advanced proposals base their security arguments on (complicated) cryptographic protocols, e.g. blind signatures or homomorphic schemes.

Canard and Traoré proposed cryptographic primitives dedicated to provide anonymous services using smart cards. Among these primitives, list signatures are specially suitable for e-voting, as they provide specific properties such as multiple vote detection. Moreover, unlike blind signatures, they do not involve a signing authority during the ballot creation process.

The purpose of this paper is to present an e-voting system, whose security relies on a tamper-resistant smart card embedding several cryptographic primitives, including list signatures.

Server-aided cryptography for anonymity (en soumission)

Portable devices (mobile phones, smart cards, ...) are very useful to access services from anywhere. However, when authentication protocols require complex cryptography, implying costly mathematical operations, these devices may become inadequate because of their limited capabilities. This is in particular the case when the device must remain anonymous and unlinkable w.r.t. the service provider since it implies the use of complex cryptographic tools. In this paper we introduce the concept of *server-aided cryptography for anonymity* by adding a powerful intermediary which helps the restricted device in its cryptographic computations. We first give a general server-aided model in this setting, which model can be applied to several cryptographic tools: group, blind and ring signatures. We also provide secure and efficient server-aided variants of well-known constructions of these schemes, and prove that the proposed variants are the best one can obtain. To the best of our knowledge, this is the first complete study on server-aided cryptography for anonymous authentications.

Low-cost cryptography for privacy in RFID systems (CARDIS 2006)

Massively deploying RFID systems while preserving people's privacy and data integrity is a major security challenge of the coming years. Up to now, it was commonly believed that, due to the very limited computational resources of RFID tags, only ad hoc methods could be used to address this problem. Unfortunately, not only those methods generally provide a weak level of security and practicality, but they also require to revise the synopsis of communications between the tag and the reader. In this paper, we give evidence that highly secure solutions can be used in the RFID environment, without substantially impacting the current communication protocols, by adequately choosing and combining low-cost cryptographic algorithms. The main ingredients of our basic scheme are a probabilistic (symmetric or asymmetric) encryption function, e.g. AES, and a coupon-based signature function, e.g. GPS. We also propose a dedicated method allowing the tag to authenticate the reader, which is of independent interest. On the whole, this leads to a privacy-preserving protocol well suited for RFID tags, which is very flexible in the sense that each reader can read and process all and only all the data it is authorized to.s

Data synchronization in privacy-preserving RFID authentication schemes (RFIDSec 2008)

Massively deploying RFID systems, while preserving people's privacy and data integrity, is a major security challenge of the coming years. This is why research related to privacy-preserving authentication is growing, including design of schemes, cryptanalysis and security models. In nearly all such schemes secret key cryptography is used, since RFID tags are extremely constrained in time and space, and untraceability is achieved by updating some data at each authentication. Unfortunately, none of them entirely resists to denial of service attacks, those in which the enemy forces updating of the tag and/or the reader by sending fake messages. Moreover, literature lacks a clear and formal way of comparing schemes w.r.t. this kind of attack. In this paper, we introduce a new characterization, called synchronizability. This allows us to, first, establish a relevant model; second, evaluate existing schemes in this model and point out their deficiencies; third, present a new scheme with all desired features.

Lighten encryption schemes for secure and private RFID systems (WLC 2010)

We provide several concrete implementations of a generic method given by Vaudenay to construct secure privacy-preserving RFID authentication and identification systems. More precisely, we give the first instantiation of the Vaudenay's result by using the IND-CCA secure DHAES cryptosystem. Next we argue that weaker cryptosystems can also be used by recalling the WIPR RFID system and giving a new protocol based on the El Gamal encryption scheme. After that, we introduce a new generic construction based on the use of any IND-CPA secure public key cryptosystem together with a MAC scheme and describe a possibility using the Hash El Gamal cryptosystem. We finally compare all these implementations, proving that, in practice, it is actually possible to use public key cryptography in privacy-preserving RFID identification systems nowadays.

Privacy-preserving RFID identification systems: model and construction (en soumission)

Privacy is commonly believed to be one of the main challenges that our society needs to face. Particularly concerning is the claimed irruption of RFID technology in our daily life. This paper presents an efficient and easy to implement privacy-preserving authentication and identification scheme suited to RFID tags. This is achieved by using both symmetric (namely a MAC scheme) and asymmetric (namely Hash El Gamal encryption) cryptography in an intricate manner. The scheme we propose achieves "strong privacy" in a new model we introduce. To the best of our knowledge, this is the most complete and easy to use model, which may be of independent interest.

Annexe C

Quelques Articles Joints

1. Complex zero-knowledge proofs of knowledge are easy to use
2. List signature schemes
3. Defeating malicious servers in a blind signatures based voting system
4. A client-side approach for privacy-preserving identity federation
5. Divisible e-cash can be truly anonymous
6. Anonymity in transferable e-cash
7. Server-aided cryptography for anonymity
8. Data synchronization in privacy-preserving RFID authentication schemes

Complex Zero-Knowledge Proofs of Knowledge are Easy to Use

Sébastien Canard, Iwen Coisel, and Jacques Traoré

Orange Labs, 42 rue des Coutures, 14000 Caen, France

Abstract. Since 1985 and their introduction by Goldwasser, Micali and Rackoff, followed in 1988 by Feige, Fiat and Shamir, zero-knowledge proofs of knowledge have become a central tool in modern cryptography. Many articles use them as building blocks to construct more complex protocols, for which security is often hard to prove. The aim of this paper is to simplify analysis of many of these protocols, by providing the cryptographers with a theorem which will save them from stating explicit security proofs. Kiayias, Tsiounis and Yung made a first step in this direction at Eurocrypt'04, but they only addressed the case of so-called "triangular set of discrete-log relations". By generalizing their result to any set of discrete-log relations, we greatly extend the range of protocols it can be applied to.

1 Introduction

The main purpose of authentication is to know who is who. More precisely, Alice wants to be convinced that the entity she communicates with is the right one. When using cryptography, this is often achieved by proving knowledge of a particular secret without (provably) revealing it. In 1985, Goldwasser, Micali and Rackoff [19] introduced the concept of zero-knowledge interactive proofs (ZKIP). The idea of using it for purposes of authentication came one year later in the article by Fiat and Shamir [15], followed in 1988 by Feige, Fiat and Shamir [14], who introduced the zero-knowledge proofs of knowledge (ZKPK).

In modern cryptography, these protocols are not only used for authentication but also as building blocks to achieve more complex purposes, such as for example guaranteeing the anonymity of a user [1, 5, 9] or committing to a secret value without being able to change one's mind [16]. In these schemes, users typically have to compute some public data relying on secret and random values, then prove that these public data are well-formed by using these building blocks. The security of the global construction relies both on the computed data and protocols they are involved in, which consequently have to be proven as being ZKPK.

The aim of this paper is to simplify analysis of many of these protocols, by providing the cryptographers with a theorem which will save them from stating explicit security proofs. Kiayias, Tsiounis and Yung made a first step in this direction at Eurocrypt'04, but they only addressed the case of so-called "triangular set of discrete-log relations". By generalizing their result to any set of discrete-log relations, we greatly extend the range of protocols it can be applied to.

1.1 Related Work

Many ZKPK have been proposed since the article of Feige et al. in 1988 [14]. When based on discrete logarithms, they are often built over a cyclic group $\mathcal{G} = \langle g \rangle$ either of known prime

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

order q (after Schnorr's article [22]) or of unknown order (but in the same range of magnitude as the order of G). In this paper, we will only consider discrete-logarithm based ZKPK in groups of unknown order, since this is the most difficult case. In this setting, the building block is the GPS authentication scheme [18], which allows to prove knowledge of a discrete logarithm in such groups.

The construction of complex cryptographic tools such as group signature schemes, credential schemes or e-cash systems, always requires more than a single proof of knowledge of a single discrete logarithm. Rather, it involves several secret values and several (discrete-log based) relations between these values. The GPS scheme has therefore to be extended in order to obtain first new building blocks as e.g. a proof of knowledge of a representation [16, 13], that involves two secret values and one relation, a proof of equality of two known representations [11, 7], which requires four secret values and two relations, or the proof that a committed value lies in an interval [4, 7, 10, 3], that necessitates several secret values and relations. Then, these various building blocks are used to construct still more elaborate protocols, the security of which must be demonstrated in detail for each of them, though the proofs are very similar to each other. As a consequence, it would be very useful to design a "general proof" which could apply to a wide range of such protocols, saving the designers from proving them secure.

Kiayias, Tsiounis and Yung [20] use such complex protocols in their construction of traceable signatures and, as an independent interest of the paper, make a first step towards designing such a general proof. They introduce the notion of *Discrete-Log Relation Set* (DLRS), that is a set of relations involving objects (as public keys and parameters) and free variables (as secret elements). For each free variable, there is a corresponding secret known by a prover $\mathcal{P}$. Then they propose a generic 3-move honest verifier zero-knowledge proof that allows $\mathcal{P}$ to prove the knowledge of these values. They also show that their construction is a ZKPK in the particular case of a triangular discrete-log relation set, that is when each relation introduces at most one new free variable w.r.t. the previous ones. They thus solve the above problem only in part, since their security proof only addresses a particular case. The aim of our paper is to solve this problem in general, for any discrete-log relation set.

1.2 Our Contribution

In this paper, we prove the soundness of any discrete-log relation set (DLRS), as defined by Kiayas, Tsiounis and Yung [20], i.e. when G is a (large) subgroup of the multiplicative group of the ring of integers modulo a composite integer. We do not address the zero-knowledge property, since it happens that it can be derived from [20] in a straight-forward manner. Unlike in [20], we do not have any restrictions on the kind of DLRS we use.

All security proofs for a ZKPK in a group of unknown order use the trick of either solving the Flexible RSA problem or retrieving all secret values involved in the proof[1]. Another contribution of this paper is that, to the best of our knowledge, our proof is the first one where the instance of the Flexible RSA problem is clearly defined.

[1] This is not the case for group of prime order.

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

1.3 Organization of the Paper

We first give some preliminaries in the next section. Section 3 introduces the first results on DLRS. It also gives evidence that the model of Kiayias *et al.* does not cover all kind of DLRS. We then give our new theorem and its proof in Section 4, then conclude in Section 5.

2 Preliminaries

In the following, G will be typically a group $QR(n)$ of quadratic residues modulo n, where n is a safe RSA modulus, as defined in the next subsection. By definition, the group G is a group of possibly unknown order but where the size of the group order, denoted by l_G, is known.

2.1 Mathematical Background

A prime p is a safe prime when $p = 2p' + 1$ and p' is a prime. A safe RSA modulus n is an integer which is the product of two distinct safe primes $p = 2p' + 1$ and $q = 2q' + 1$, that is $n = pq$. The following technical lemma (see e.g. [17]) will be useful.

Lemma 1. *Let $n = pq$, where $p < q$, $p = 2p' + 1$, $q = 2q' + 1$, and p, q, p', q' are all prime numbers. Then,*

1. *The order of elements in $\mathbb{Z}_n^*$ is in $\{1, 2, p', q', 2p', 2q', p'q', 2p'q'\}$.*
2. *Given an element $w \in \mathbb{Z}_n^* \setminus \{-1, 1\}$ such that $ord(w) < p'q'$, then either $gcd(w - 1, n)$ or $gcd(w + 1, n)$ is a prime factor of n.*

As a consequence of the above lemma, any value found by a party that does not know (and cannot compute) the factorization of n must be of order at least $p'q'$ in $\mathbb{Z}_n^*$ (except for -1 and 1).

Lemma 2. *Let $n = pq$, where $p < q$, $p = 2p' + 1$, $q = 2q' + 1$, and p, q, p', q' are all prime numbers.*

If $\nu^2 = 1$ and $\nu \in QR(n)$ then $\nu = 1$.

Proof. As a safe modulus, n is also a Blum number (a product of two primes equal to $3 mod 4$). As a consequence, any element of $QR(n)$ has exactly one square root in $QR(n)$. Since 1 is in $QR(n)$, 1 is the only square root of 1 in $QR(n)$.

2.2 Number Theoretic Assumption

The security of discrete-logarithm based zero-knowledge proofs of knowledge in groups of unknown order relies on the Flexible RSA assumption (independently introduced by Barić and Pfitzmann [2] and by Fujisaki and Okamoto [16], also known as Strong RSA). This assumption can be stated as follows, restricted to safe modulus, as it is the case in our paper.

Assumption 1 (Flexible RSA) *Given a safe RSA modulus n and $\Gamma \in QR(n)$, it is infeasible to find $u \in \mathbb{Z}_n^*$ and $e \in \mathbb{Z}_{>1}$ such that $u^e = \Gamma \pmod n$, in time polynomial in $\lceil \log p'q' \rceil$ with a non-negligible probability.*

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

2.3 Zero-Knowledge Proofs of Knowledge

The notion of interactive zero-knowledge proof of knowledge has been formalized by Feige, Fiat and Shamir [14]. As in [20], we only consider honest verifier zero-knowledge since this is always the considered setting in studied complex constructions. Let us give the following (informal) definition.

Definition 1. *An interactive protocol between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$, that takes on input $\mathcal{Y}$, is a zero-knowledge proof of knowledge of a secret x if the three following properties are verified.*

- *Completeness: given an honest prover $\mathcal{P}$ and an honest verifier $\mathcal{V}$, the protocol succeeds with overwhelming probability.*
- *Soundness: given a dishonest prover $\tilde{\mathcal{P}}$ that is accepted by a verifier $\mathcal{V}$ with non-negligible probability, it is possible to construct a probabilistic polynomial time Turing machine $\mathcal{M}$ that can find x by interacting with $\tilde{\mathcal{P}}$.*
- *(Honest verifier) zero-knowledge: it exists a probabilistic polynomial-time Turing machine that takes on input $\mathcal{Y}$ and which can simulate the communications between an honest prover $\mathcal{P}$ and an honest verifier $\mathcal{V}$ such that these simulated communications are indistinguishable from those between a real prover $\mathcal{P}$ and a real honest verifier $\mathcal{V}$.*

3 First Result on DLRS

Discrete-log relation sets (DLRS) were introduced by Kiayias *et al.* [20], and are useful when constructing complex proofs of knowledge for protocols operating over any group, even of unknown order. These constructions are quite useful in many complex cryptographic protocols [16, 1, 5, 9].

3.1 Introduction of the Concept of DLRS

The following definition of a DLRS has been proposed in [20]:

Definition 2. *(see [20]) Let G be a finite group. A discrete-log relation set R with z relations over r variables and m objects is a set of relations defined over the objects $A_1, \ldots, A_m \in G$ and the free variables $\alpha_1, \ldots, \alpha_r$ with the following specifications:*

1. *the i-th relation in the set R is specified by a tuple $\langle a_1^i, \ldots, a_m^i \rangle$ so that each a_j^i is selected to be one of the free variables $\{\alpha_1 \ldots, \alpha_r\}$ or an element of $\mathbb{Z}$. The relation is to be interpreted as $\prod_{j=1}^{m} A_j^{a_j^i} = 1$.*
2. *every free variable α_ω is assumed to take values in a finite integer range $]2^{l_\omega} - 2^{\mu_\omega}, 2^{l_\omega} + 2^{\mu_\omega}[$ where $l_\omega, \mu_\omega \geq 0$.*

We will write $R(\alpha_1, \ldots, \alpha_r)$ to denote the conjunction of all relations $\prod_{j=1}^{m} A_j^{a_j^i} = 1$ that are included in R.

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

Notation. The following notation will be used for the rest of the article. For the i-th relation, we define for each free variable α_ω ($\omega \in \{1, \ldots, r\}$) the set $\mathcal{J}_{\omega,i} \subseteq \{1, \ldots, m\}$ of the variable's locations in the tuple $\langle a_1^i, \ldots, a_m^i \rangle$. If a free variable α_ω is not contained in the relation i, the set $\mathcal{J}_{\omega,i}$ is empty. We also set $\mathcal{J}_i = \bigcup_{\omega=1}^{r} \mathcal{J}_{\omega,i}$. Note that $j \notin \mathcal{J}_i$ means $a_j^i \in \mathbb{Z}$. Finally, for all $\omega = 1, \ldots, r$, let us denote $\tilde{A}_{\omega,i} = \prod_{j \in \mathcal{J}_{\omega,i}} A_j$. Naturally, if $\mathcal{J}_{\omega,i} = \phi$ then $\tilde{A}_{\omega,i} = 1$. Consequently, the i-th relation verifies the following relation.

$$\prod_{j=1}^{m} A_j^{a_j^i} = 1 \;\Leftrightarrow\; \prod_{\omega=1}^{r} \tilde{A}_{\omega,i}^{\alpha_\omega} \prod_{j \notin \mathcal{J}_i} A_j^{a_j^i} = 1$$

Using these notations, a 3-move honest verifier zero-knowledge proof allows a prover that knows witnesses $x_1, \ldots, x_r$ such that $\forall \omega, x_\omega \in]2^{l_\omega} - 2^{\epsilon(\mu_\omega+k)+2}, 2^{l_\omega} + 2^{\epsilon(\mu_\omega+k)+2}[$ and $R(x_1, \ldots, x_r) = 1$ to prove knowledge of these values, is presented in [20] and shown in Figure 1, where ϵ and k are both security parameters such that $\epsilon > 1$ and $k \in \mathbb{N}$.

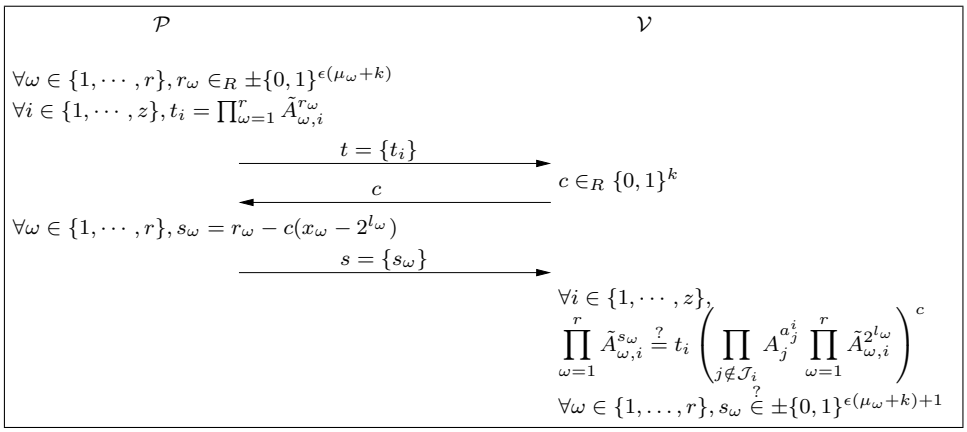

Fig. 1. Discrete-log Relation Set R

Remark 1. Note that the proof of knowledge of Figure 1 only proves that a witness $x \in]2^l - 2^\mu, 2^l + 2^\mu[$ lies in $]2^l - 2^{\epsilon(\mu+k)+2}, 2^l + 2^{\epsilon(\mu+k)+2}[$. If needed, Boudot presents in [4] a scheme that provides a perfect proof but with less efficiency. If the interval is small, it is also possible to use a bit-by-bit solution, such as in [3, 8].

3.2 The Result of Kiayias, Tsiounis and Yung

In [20], the authors present a particular case of our result. They prove the security of the construction of DLRS R presented in Figure 1 w.r.t. Definition 1 (see Section 2.3) in the case the relation R is *triangular*, and when G is the group $QR(n)$ of quadratic residue modulo n where n is a safe RSA modulus. In the following, G will also be this group. In the next section, we will prove the security of this construction in the general case. A triangular DLRS is introduced in [20] by the following definition.

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

Definition 3. *(see [20]) A discrete-log relation set R is* triangular *if for each relation i containing the $b+1$ free variables $\alpha_\omega, \alpha_{\omega_1}, \ldots, \alpha_{\omega_b}$ it holds that $\{\alpha_{\omega_1}, \ldots, \alpha_{\omega_b}\}$ is a subset of the union of all the free variables involved in relations $1, \ldots, i-1$.*

In this context, Kiayias *et al.* prove that the construction in Figure 1 is secure, i.e. for any triangular discrete-log relation set R the 3-move protocol of figure 1 is complete, sound and honest-verifier zero-knowledge.

3.3 On the Use of Kiayias, Tsiounis and Yung Result

If a complex proof of knowledge can be represented by a triangular discrete-log relation set, the construction of [20] is suitable. This is for example the case in the group signature scheme proposed by Ateniese et al. [1], where the DLRS is composed of the 9 objects $T_1, T_2, T_3, A, a_0, a, y, g, h$, the 4 free-variables $\alpha, \beta, \gamma, \delta$ such that the 4 relations $a_0 = T_1^\alpha/(a^\beta y^\gamma) \wedge T_2 = g^\delta \wedge 1 = T_2^\alpha/g^\gamma \wedge T_3 = g^\alpha h^\delta)$ are verified in order to produce a signature.

But, in some cases, their approach cannot be applied. For example, the construction of [5] uses a DLRS with 8 objects $(C, C_1, C_2, C_3, g, h, 1/g, 1/h)$ and 11 variables $(\alpha, \beta, \gamma, \delta, \eta, \zeta, \phi, \psi, \theta, \sigma, \nu)$ verifying the following conjunction of the 7 relations

$$C = g^\alpha h^\phi \wedge g = \left(\frac{C}{g}\right)^\gamma h^\psi \wedge g = (gC)^\sigma h^\nu \wedge C_3 = g^\zeta h^\eta$$
$$\wedge C_1 = g^\alpha h^\theta \wedge v = C_2^\alpha \left(\frac{1}{h}\right)^\beta \wedge 1 = C_3^\alpha \left(\frac{1}{h}\right)^\delta \left(\frac{1}{g}\right)^\beta.$$

This DLRS clearly cannot be represented by a triangular discrete-log relation set.

This is also the case for [9] and more simply if Alice wants to commit to the value x using the Fujisaki-Okamoto construction [16], and that she knows the commited value. The latter can be done by computing $PK(\alpha, \beta : C = g^\alpha h^\beta)$, that is a DLRS R of 1 relation over 2 variables and 3 objects.

Consequently, there is sometimes more than one new free-variable at each new relation. More generally speaking, when a discrete-log relation set R is not triangular, then for each relation i containing the free variables $\alpha_{\tilde{\omega}_1}, \ldots, \alpha_{\tilde{\omega}_d}, \alpha_{\omega_1}, \ldots, \alpha_{\omega_b}$ it holds that the free variables $\alpha_{\omega_1}, \ldots, \alpha_{\omega_b}$ were contained in the union of all the free variables involved in relations $1, \ldots, i-1$. But that does not imply that the construction proposed in Figure 1 does not suit the general case. What lacks is a security proof for this construction in the general setting: the result of Kiayias *et al.* [20] cannot be used as it is in the general case.

4 Generalization of the DLRS Theorem

In the general setting, the proof of completeness and honest-verifier zero-knowledge are not different to the one described in [20]. They will consequently not be treated in this paper. On the contrary, the proof of soundness of [20] must be deeply modified to suit the model considering any kind of DLRS, not only the triangular ones. This adaptation is the actual contribution of this paper.

An interactive protocol between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$ verifies the soundness property if a dishonest prover $\tilde{\mathcal{P}}$ can not be accepted by a verifier $\mathcal{V}$ with non-negligible probability. Generally, a probabilistic polynomial time Turing machine $\mathcal{M}$ that can find x by interacting with $\tilde{\mathcal{P}}$ is constructed to prove this property.

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

4.1 Our Result in a Nutshell

In this section, we briefly present our proof of soundness for all kinds of DLRS. The global structure of our proof is described in Figure 2.

In the first step, we assume that there exists $\tilde{\mathcal{P}}$ able to produce, with non-negligible probability, valid proofs of knowledge without knowing the secret values $X = \{x_1, \ldots, x_s\}$. Our aim is to construct a p.p.t. Turing machine $\mathcal{M}$ which, for each equation, is able to solve a given instance of the Flexible RSA problem (FRSA).

We first give an instance (n, Γ) of the Flexible RSA problem to $\mathcal{M}$. $\mathcal{M}$ generates a random DLRS R, function of this instance. We then ask $\tilde{\mathcal{P}}$ to produce a valid proof of knowledge until we obtain two valid conversations $\langle t, c, s\rangle, \langle t, c^*, s^*\rangle$, where $c \neq c^*, t = \{t_1, \ldots, t_z\}, s = \{s_1, \ldots, s_r\}, s^* = \{s_1^*, \ldots, s_r^*\}$. We also denote $\tilde{s}_i = s_i - s_i^*$ for all i, $\tilde{S} = \{\tilde{s}_1, \ldots, \tilde{s}_r\}$ and $\tilde{c} = c - c^*$.

From these relations, $\mathcal{M}$ then computes for each of the z relations an independent equation only depending on c, c^*, s and s^*. Each couple (s_i, s_i^*) is related to a free variable, and thus to a secret. Our aim is then to retrieve the value of all secrets.

In a similar way to [20], the machine $\mathcal{M}$ always operates as follows.

1. For each of the z relations, it first pushes aside the couples (s_i, s_i^*) for which the secret has already been retrieved. This step is not done for the first relation.
2. It then calculates the number of secrets that are unknown in the relation. Depending on it, there are three cases.
   (a) There is only one unknown secret. This is the case that has been studied in [20]. In fact, if, for each relation, there is only one unknown secret, the DLRS is then triangular. The conclusion is that either we can compute all secret or we can solve the instance (n, Γ) of the Flexible RSA problem.
   (b) There are two unknown secrets. This case corresponds to the ZKPK of a representation. In a group of unknown order, the case has been studied in [13], using the Root assumption. We thus adapt it by using the Flexible RSA assumption. The conclusion is that either we can compute all secrets or we can solve the instance (n, Γ) of the Flexible RSA problem.
   (c) The general case (up to three but the cases 1 and 2 can also be seen as particular cases) is the one we study in this paper. The relation can thus be denoted as $\tilde{A}_1^{\tilde{s}_1} \ldots \tilde{A}_d^{\tilde{s}_d} = \Psi_i^{\tilde{c}}$. $\tilde{A}_1, \ldots, \tilde{A}_d$ correspond to the objects defined after the DLRS definition (see Section 3) and Ψ_i is the product of a constant element and possibly some objects $\tilde{\mathcal{A}}_j$ raised to the power of secret values already compute. $\tilde{c}, \tilde{S}$ are dependant of c, c^*, S, S^*.
   We then study two cases. In the first one, $\mathcal{M}$ retrieves all secrets involved in this relation. The second case is also divided into two possible cases.
      i. $\mathcal{M}$ can solve the instance (n, Γ) of the FRSA problem.
      ii. We prove that the second case only happens with probability less than 1/2.

   If $\mathcal{M}$ is able to find all the secret values, $\tilde{\mathcal{P}}$ can also do it. So, under the assumption that $\tilde{\mathcal{P}}$ does not know these values, we conclude that $\mathcal{M}$ solves the given instance of the Flexible RSA problem.

In all papers where there is a ZKPK in the group of unknown order $QR(n)$, such as in the paper of Kiayias, Tsiounis and Yung [20] but also e.g. in [1, 6], a p.p.t. Turing machine $\mathcal{M}$ is constructed so as to solve with a non-negligible probability an instance of the Flexible RSA

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

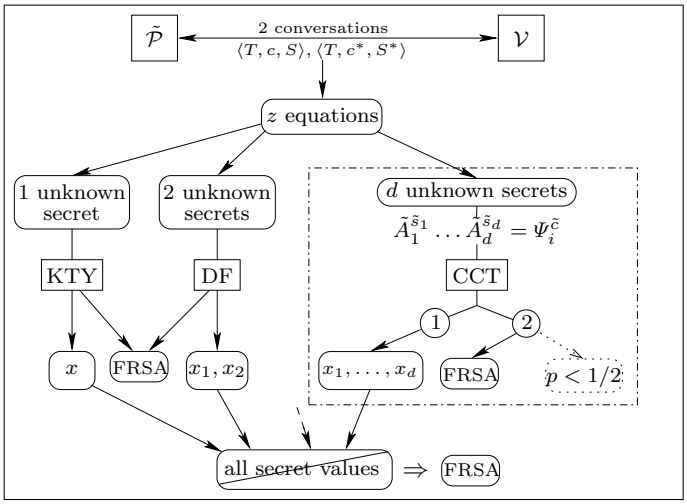

Fig. 2. Sketch of proof

problem. However, this instance is never specified so that it could possibly be an easy instance of the problem.

More precisely, the solved instance corresponds to the modular multiplication of public parameters (the A_i's) but nothing is said about the difficulty of solving the Flexible RSA on one A_i nor on the modular multiplication of some of them. It seems better, and that's what we do in our proof, to introduce a challenger $\mathcal{C}$ which gives to $\mathcal{M}$ a random instance of the Flexible RSA problem at the beginning of the proof.

Nevertheless, as we will see in our proof, $\mathcal{M}$ will need to interact possibly with several dishonest provers $\tilde{\mathcal{P}}$, depending on the objects $A_1, \ldots, A_m$ the machine $\mathcal{M}$ has to use to solve the Flexible RSA instance. The number z of relations and the number r of free variables can be unchanged between all the interactions. This consequently implies the use of an attacker $\tilde{\mathcal{P}}$ being able to break the soundness of a DLRS for a polynomial number of tuples $A_1, \ldots, A_m$.

4.2 The New Theorem

We can then introduce our new theorem and prove the security of the construction in Figure 1 in the case of any discrete-log relation set.

Theorem 1. *Let $G = QR(n)$ where $n(= (2p' + 1)(2q' + 1))$ is safe. For any discrete-log relation set R the 3-move protocol of Figure 1 is a honest-verifier zero-knowledge proof of knowledge that can be used by a first party (prover) knowing a witness for R to prove knowledge of the witness to a second party (verifier).*

Proof. We have to prove that the protocol of Figure 1 verifies the three properties of completeness, soundness and honest verifier zero-knowledge. The proof of completeness and honest verifier zero-knowledge can be found in [20]. They will not be treated in this proof. The proof of soundness of [20] must be modified to suit our model (all kinds of DLRS, not only the triangular ones).

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

Assume it exists a dishonest prover $\tilde{\mathcal{P}}$ attacking the soundness of the protocol presented in Figure 1. It means that $\tilde{\mathcal{P}}$ is able to produce valid conversations for this protocol with non-negligible probability, and without knowing all the involved secrets. We define a p.p.t. Turing machine $\mathcal{M}$ which solves a given instance of the Flexible RSA problem, using $\tilde{\mathcal{P}}$ as an oracle. Let $\mathcal{C}$ be the challenger who gives the instance (n, Γ) of the Flexible RSA problem to $\mathcal{M}$. The Turing machine $\mathcal{M}$:

- takes on input the instance (n, Γ) of the FRSA problem given by $\mathcal{C}$,
- generates a random DLRS R,
- interacts with $\tilde{\mathcal{P}}$,
- solves the given instance using $\tilde{\mathcal{P}}$'s outputs.

In order to define R, $\mathcal{M}$ randomly chooses integers $\gamma_\omega \in \{1, \ldots, n^2\}$ and computes $A_\omega = \Gamma^{\gamma_\omega}$, for $\omega \in \{1, \ldots, m\}$. Under the factorisation assumption, the order of Γ is $\phi(n)/4$ and consequently, the A_ω are distributed over $QR(n)$. $\mathcal{M}$ sends R to the dishonest prover $\tilde{\mathcal{P}}$. Let $\langle t_1, \ldots, t_z, c, s_1, \ldots, s_r \rangle$ and $\langle t_1, \ldots, t_z, c^*, s_1^*, \ldots, s_r^* \rangle$, with $c \neq c^*$, be two accepted protocols for R between $\tilde{\mathcal{P}}$ and an (honest) verifier. As these protocols are valid, both following relations are true for all $i \in \{1, \ldots, z\}$:

$$\prod_{\omega=1}^{r} \tilde{A}_{\omega,i}^{s_\omega} = t_i \left(\prod_{j \notin \mathcal{J}_i} A_j^{a_j^i} \prod_{\omega=1}^{r} \tilde{A}_{\omega,i}^{2^{l_\omega}} \right)^c \text{ and } \prod_{\omega=1}^{r} \tilde{A}_{\omega,i}^{s_\omega^*} = t_i \left(\prod_{j \notin \mathcal{J}_i} A_j^{a_j^i} \prod_{\omega=1}^{r} \tilde{A}_{\omega,i}^{2^{l_\omega}} \right)^{c^*}$$

$$\Rightarrow \prod_{\omega=1}^{r} \tilde{A}_{\omega,i}^{s_\omega - s_\omega^*} = \left(\prod_{j \notin \mathcal{J}_i} A_j^{a_j^i} \prod_{\omega=1}^{r} \tilde{A}_{\omega,i}^{2^{l_\omega}} \right)^{c-c^*}. \quad (1)$$

The proof consists now in proving that using relations (1) for all $i \in \{1, \ldots, z\}$, $\mathcal{M}$ is able to solve the given instance of the Flexible RSA problem. First, we introduce the notations we will use in the following of the proof. For $\omega \in \{1, \ldots, r\}$: $\tilde{s}_\omega := s_\omega - s_\omega^*$, and $\tilde{c} := c - c^*$. We also introduce the sets of distinct integers $\Omega_i = \{\omega_{i,1}, \ldots, \omega_{i,d}\}$, for each relation i (i.e. for i from 1 to z), such that the free variables $\alpha_{\omega_{i,1}}, \ldots, \alpha_{\omega_{i,d}}$ are the ones involved in the i-th relation. Using these notations, for $i \in \{1, \ldots, z\}$, the relation (1) can be written:

$$\prod_{\omega \in \Omega_i} \tilde{A}_{\omega,i}^{\tilde{s}_\omega} = \left(\prod_{j \notin \mathcal{J}_i} A_j^{a_j^i} \prod_{\omega \in \Omega_i} \tilde{A}_{\omega,i}^{2^{l_\omega}} \right)^{\tilde{c}}. \quad (2)$$

Relation 1. Considering the first relation, there are two cases:

- $\tilde{c}$ divides all the integers $\tilde{s}_\omega$
  The particular case where $d = 1$ (as in [20]) is included in the general case. So we restrict our proof to the general case, where $d \geq 1$. It holds that the first relationship in R involves d free variables denoted by α_ω for $\omega \in \Omega_1 = \{\omega_{1,1}, \ldots, \omega_{1,d}\}$. In this case, we have the following relation, where $\tilde{A}_\omega$ stands for $\tilde{A}_{\omega,1}$:

$$\prod_{\omega \in \Omega_1} \tilde{A}_{\omega}^{\tilde{s}_\omega} = \left(\prod_{\omega \in \Omega_1} \tilde{A}_{\omega}^{2^{l_\omega}} \prod_{j \notin \mathcal{J}_1} A_j^{a_j^1} \right)^{\tilde{c}}.$$

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

As $\tilde{c}$ divides $\tilde{s}_\omega$, for all $\omega \in \Omega_1$, the previous relation becomes (see remark below):

$$\prod_{\omega\in\Omega_1} \tilde{A}_\omega^{\frac{-\tilde{s}_\omega}{\tilde{c}}+2^{l_\omega}} \prod_{j\notin\mathcal{J}_1} A_j^{a_j^1} = 1. \tag{3}$$

Remark 2. In fact, we have the following equivalence :

$$\prod_{\omega\in\Omega_1} \tilde{A}_\omega^{\tilde{s}_\omega} = \left(\prod_{\omega\in\Omega_1} \tilde{A}_\omega^{2^{l_\omega}} \prod_{j\notin\mathcal{J}_1} A_j^{a_j^1}\right)^{\tilde{c}} \Leftrightarrow \prod_{\omega\in\Omega_1} \tilde{A}_\omega^{\frac{\tilde{s}_\omega}{\tilde{c}}} = \nu \prod_{\omega\in\Omega_1} \tilde{A}_\omega^{2^{l_\omega}} \prod_{j\notin\mathcal{J}_1} A_j^{a_j^1},$$

with $\nu^c = 1$. Indeed, by definition $\tilde{c} < 2^k$ and thus $\tilde{c} < min(p, q)$. By Lemma 1, we can then affirm that the order of ν can only be equal to 1 or 2 and by lemma 2, that ν can only be equal to 1. We will not repeat this remark later, even when it holds.

The equality 3 implies that we have constructed the d witnesses for each ω-th variable $\tilde{x}_\omega = \frac{\tilde{s}_\omega}{\tilde{c}} + 2^{l_\omega} = \frac{s_\omega - s_\omega^*}{c-c^*} + 2^{l_\omega}$ where $\omega \in \Omega_1$.
We verify that these values are in the right interval. For $\omega \in \Omega_1$, $\tilde{s}_\omega \in \pm\{0,1\}^{\epsilon(\mu_\omega+k)+2}$ (since $s_\omega, s_\omega^* \in \pm\{0,1\}^{\epsilon(\mu_\omega+k)+1}$, it implies that $s_\omega^* - s_\omega \in \pm\{0,1\}^{\epsilon(\mu_\omega+k)+2}$) it follows that $\frac{\tilde{s}_\omega}{\tilde{c}} \in \pm\{0,1\}^{\epsilon(\mu_\omega+k)+2}$ and as a result $\tilde{x}_\omega \in]2^{l_\omega} - 2^{\epsilon(\mu_\omega+k)+2}, 2^{l_\omega} + 2^{\epsilon(\mu_\omega+k)+2}[$. Consequently, $\mathcal{M}$ finds the secrets $\{\tilde{x}_\omega\}$ *for* $\omega \in \Omega_1$ in polynomial time, $\tilde{\mathcal{P}}$ can also find it. So we can assume that $\tilde{\mathcal{P}}$ already knows it.

- It exists at least one integer $\omega \in \Omega_1$ such that $\tilde{c}$ does not divide $\tilde{s}_\omega$.
  Now, we prove that $\mathcal{M}$ solves the given instance (n, Γ) of the FRSA problem on G. Let

$$T_1 = \left(\prod_{\omega\in\Omega_1} \tilde{A}_\omega^{2^{l_\omega}} \prod_{j\notin\mathcal{J}_1} A_j^{a_j^1}\right).$$

For all j in $\{1, \ldots, d\}$, $A_j = \Gamma^{\gamma_j}$, and for all $\omega \in \Omega_1$, we have $\tilde{A}_\omega = \prod_{j\in\mathcal{J}_{\omega,1}} A_j = \prod_{j\in\mathcal{J}_{\omega,1}} \Gamma^{\gamma_j} = \Gamma^{\sum_{j\in\mathcal{J}_{\omega,1}} \gamma_j}$. We define $\theta_\omega = \sum_{j\in\mathcal{J}_{\omega,1}} \gamma_j \pmod{n^2}$ for all $\omega \in \Omega_1$. Consequently, with those notations relation (2) becomes:

$$\prod_{\omega\in\Omega_1} \left(\Gamma^{\sum_{j\in\mathcal{J}_{\omega,1}} \gamma_j}\right)^{\tilde{s}_\omega} = T_1^{\tilde{c}} \quad \Leftrightarrow \quad \Gamma^{\sum_{\omega\in\Omega_1} \theta_\omega \tilde{s}_\omega} = T_1^{\tilde{c}}. \tag{4}$$

Without loss of generality, we assume that integers $\tilde{s}_{1,1}, \ldots, \tilde{s}_{1,d_1}$ are divisible by $\tilde{c}$, as opposed to integers $\tilde{s}_{1,d_1+1}, \ldots, \tilde{s}_{1,d_2}$, with $1 \le d_1 < d_2 = d$. If $d_2 = 1$, because we assumed that $\tilde{c}$ does not divide all the $\tilde{s}_\omega$, then $d_1 = 0$.
Then there are two cases:

1. If $\tilde{c}$ does not divide $\sum_{\omega\in\Omega_1} \theta_\omega \tilde{s}_\omega$, $\mathcal{M}$ can solve the given instance of the Flexible RSA problem as follows. Let δ be the greatest common divisor of $\tilde{c}$ and $\sum_{\omega\in\Omega_1} \theta_\omega \tilde{s}_\omega$. There exist α and β in $\mathbb{Z}$ such that $\alpha\tilde{c} + \beta\big(\sum_{\omega\in\Omega_1} \theta_\omega \tilde{s}_\omega\big) = \delta$. It follows that

$$\Gamma = \Gamma^{\left(\alpha\tilde{c} + \beta(\sum_{\omega\in\Omega_1} \theta_\omega \tilde{s}_\omega)\right)/\delta} = (\Gamma^\alpha T_1^\beta)^{\tilde{c}/\delta}.$$

   By assumption, $\delta < \tilde{c}$ and so, we can set $e = \tilde{c}/\delta$ and $u = \Gamma^\alpha T_1^\beta$, which is a solution of the Flexible RSA problem on G relatively to the instance (n, Γ).

   Remark 3. This part of the proof works with any values of the integer $d_1 < d_2$.

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

2. If $\tilde{c}$ divides $\sum_{\omega\in\Omega_1}\theta_\omega\tilde{s}_\omega$, we prove that, as $\tilde{\mathcal{P}}$ does not have complete information about the θ_ω's, this case only happens with probability less or equal to 1/2. Consequently, case (1) happens with probability greater than 1/2 and the probability to break the Flexible RSA assumption is greater than 1/2. The strategy consists in choosing the θ_ω's until we get back on case (1). This quickly happens in a bounded time with non-negligible probability.
   Let f be a prime factor of $\tilde{c}$ and e an integer such that:
   – f^e is the greatest power of f that divides $\tilde{c}$,
   – at least one of the $\tilde{s}_\omega$ is non-zero modulo f^e.
   This value must exist since $\tilde{c}$ does not divide at least one of the $\tilde{s}_\omega$, even if $d_2 = 1$. For all $\omega \in \Omega_1$, we define $b_\omega = \theta_\omega \pmod{\mathrm{ord}(G)}$ and h_ω such that $\theta_\omega = b_\omega + h_\omega\,\mathrm{ord}(G)$. Note that the $\tilde{A}_{\omega,1}$'s represent all the information the machine $\tilde{\mathcal{P}}$ knows about the θ_ω's and the b_ω's are uniquely determined from the $\tilde{A}_{\omega,1}$'s, whereas the h_ω's are completely unknown. As f^e divides $\sum_{\omega\in\Omega_1}\theta_\omega\tilde{s}_\omega$ (since $\tilde{c}$ does), it follows that

$$\sum_{\omega\in\Omega_1}\theta_\omega\tilde{s}_\omega = 0 \pmod{f^e} \text{ and } \sum_{j=1}^{d_2}\theta_{\omega_{1,j}}\tilde{s}_{\omega_{1,j}} = 0 \pmod{f^e}.$$

   We know that for j from 1 to d_1, $\tilde{s}_{\omega_{1,j}} \equiv 0 \pmod{f^e}$ as they are divisible by $\tilde{c}$, consequently, $\sum_{j=1}^{d_1}\theta_{\omega_{1,j}}\tilde{s}_{\omega_{1,j}} \equiv 0 \pmod{f^e}$.

$$\sum_{j=d_1+1}^{d_2} b_{\omega_{1,j}}\tilde{s}_{\omega_{1,j}} + \mathrm{ord}(G)\sum_{j=d_1+1}^{d_2} h_{\omega_{1,j}}\tilde{s}_{\omega_{1,j}} = 0 \pmod{f^e}. \tag{5}$$

   Since $f^e \leq 2^k \leq min(p', q'$, we have $|G| \neq 0 \pmod{f}$. $\tilde{\mathcal{P}}$ does not know anything about the h_ω's except that they follow the uniform distribution and that they satisfy equation (5). Let $\tilde{\omega}$ be one of the indexes such that $\tilde{s}_{\tilde{\omega}}$ is not divisible by f^e. If $d_2 = 1$, it is evident that $\tilde{\omega} = 1$. If we fix the h_ω's for $\omega \in \Omega_1/\{\tilde{\omega}\}$, then the number of solutions modulo f^e of the equation (5) is at most $gcd(|G|\tilde{s}_{\tilde{\omega}}, f^e)$. This number is necessarily a power of f, since f^e does not divide $|G|\tilde{s}_{\tilde{\omega}}$, and at most f^{e-1}. Since for all $\omega \in \Omega_1$, θ_ω has been chosen from a large interval, the distribution of b_ω is statistically indistinguishable from the uniform distribution on $\mathbb{Z}_{p'q'}$. Moreover the distribution of h_ω is statistically indistinguishable from the uniform distribution on $\{0, \ldots, M\}$, where $M = \lfloor n^2/p'q' \rfloor$. Thus, there are nearly M^{d_2} possible tuples $\langle h_1, \ldots, h_{d_2}\rangle$ uniformly distributed [12]. Let $\mathsf{w} \in \mathbb{R}$ such that $M = \mathsf{w}f^e$. The number of solutions of the equation is at most $[\mathsf{w}f^{e-1}]M^{d_2-1}$, hence the probability that the h_ω's verify the equation is at most

$$\frac{[\mathsf{w}f^{e-1}]M^{d_2-1}}{M^{d_2}} \leq \frac{\mathsf{w}f^{e-1}}{M} \leq \frac{\mathsf{w}f^{e-1}}{\mathsf{w}f^e} \leq \frac{1}{f} \leq \frac{1}{2}$$

   We can then solve the instance of the Flexible RSA problem with non-negligible probability.

If $\tilde{\mathcal{P}}$ outputs integers $\tilde{c}, \tilde{s}_1, \ldots, \tilde{s}_r$ such that relation (4) is verified and at least one of the $\tilde{s}_\omega$ is not divisible by $\tilde{c}$, for $\omega \in \Omega_1$, then $\mathcal{M}$ solves the given instance of the Flexible RSA problem.

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

Relation i. Now, we assume that we have processed all the relations with index less than i and $\mathcal{M}$ did not already solve the instance of the FRSA problem. We process the i-th relation which involves variables α_ω, for all $\omega \in \Omega_i (= \{\omega_{i,1}, \ldots, \omega_{i,d}\})$. As we have processed all the relations with index less than i, some of these variables are already known. We split Ω_i in two sets of integers $\Omega_{i,1} = \{\omega_{i,1}, \ldots, \omega_{i,d_2}\}$ and $\Omega_{i,2} = \{\omega_{i,d_2+1}, \ldots, \omega_{i,d}\}$ so that the variables α_ω, for $\omega \in \Omega_{i,2}$ are already contained in previous relations. We assume that these variables are known by $\mathcal{M}$ and then by $\tilde{\mathcal{P}}$. By an inductive argument, we construct witnesses for the free-variables $\tilde{x}_\omega = \frac{-\tilde{s}_\omega}{\tilde{c}} + 2^{l_\omega} = \frac{s^*_\omega - s_\omega}{c - c^*} + 2^{l_\omega}$, and $\tilde{c}$ divides $\tilde{s}_\omega$, for all $\omega \in \Omega_{i,2}$. There are again two cases:

- $\tilde{c}$ divides $\tilde{s}_\omega$, for all $\omega \in \Omega_{i,1}$
  First, we study the particular case where $d_2 = 1$ (see also [20]): the i-th relation in R involves variables $\alpha_{\omega_{i,1}}, \ldots, \alpha_{\omega_{i,d}}$, where $\alpha_{\omega_{i,1}}$ is the only one for which the witness associated is not yet constructed. Using relation (2), the i-th relation becomes, where $\tilde{A}_\omega$ stands for $\tilde{A}_{\omega,i}$:

$$\tilde{A}_{\omega_{i,1}}^{\tilde{s}_{\omega_{i,1}}} \prod_{\omega \in \Omega_{i,2}} \tilde{A}_\omega^{\tilde{s}_\omega} = \left(\tilde{A}_{\omega_{i,1}}^{2^{l_{\omega_{i,1}}}} \prod_{\omega \in \Omega_{i,2}} \tilde{A}_\omega^{2^{l_\omega}} \prod_{j \notin \mathcal{J}_i} A_j^{a_j^i} \right)^{\tilde{c}}$$

$$\tilde{A}_{\omega_{i,1}}^{\tilde{s}_{\omega_{i,1}}} = \left(\tilde{A}_{\omega_{i,1}}^{2^{l_{\omega_{i,1}}}} \prod_{\omega \in \Omega_{i,2}} \tilde{A}_\omega^{\tilde{x}_\omega} \prod_{j \notin \mathcal{J}_i} A_j^{a_j^i} \right)^{\tilde{c}}.$$

  As $\tilde{c}$ divides $s_{\omega_{i,1}}$ we obtain the following relation :

$$\tilde{A}_{\omega_{i,1}}^{\frac{-\tilde{s}_{\omega_{i,1}} + 2^{l_{\omega_{i,1}}}}{\tilde{c}}} \prod_{\omega \in \Omega_{i,2}} \tilde{A}_\omega^{x_\omega} \prod_{j \notin \mathcal{J}_i} A_j^{a_j^i} = 1.$$

  The above equality implies that we have constructed the witness for the variables $\tilde{x}_{\omega_{i,1}} = \frac{-\tilde{s}_{\omega_{i,1}}}{\tilde{c}} + 2^{l_{\omega_{i,1}}} = \frac{s^*_{\omega_{i,1}} - s_{\omega_{i,1}}}{c - c^*} + 2^{l_{\omega_{i,1}}}$. As previously, it is possible to show that this witness is in the right interval, i.e. $\tilde{x}_{\omega_{i,1}} \in]2^{l_{\omega_{i,1}}} - 2^{\epsilon(\mu_{\omega_{i,1}}+k)+2}, 2^{l_{\omega_{i,1}}} + 2^{\epsilon(\mu_{\omega_{i,1}}+k)+2}[$. We can also assume in this case that $\tilde{\mathcal{P}}$ already knows this witness.
  Now, we study the general case where $d_2 \neq 1$: the i-th relation in R involves variables $\alpha_{\omega_1}, \ldots, \alpha_{\omega_d}$ so that variables $\alpha_{\omega_{d_2+1}}, \ldots, \alpha_{\omega_d}$ were already contained in previous relations. So the associated witnesses are known by $\tilde{\mathcal{P}}$. Using relation (2), the i-th relation becomes:

$$\prod_{\omega \in \Omega_{i,1}} \tilde{A}_\omega^{\tilde{s}_\omega} \prod_{\omega \in \Omega_{i,2}} \tilde{A}_\omega^{\tilde{s}_\omega} = \left(\prod_{\omega \in \Omega_{i,1}} \tilde{A}_\omega^{2^{l_\omega}} \prod_{\omega \in \Omega_{i,2}} \tilde{A}_\omega^{2^{l_\omega}} \prod_{j \notin \mathcal{J}_i} A_j^{a_j^i} \right)^{\tilde{c}} \tag{6}$$

$$\prod_{\omega \in \Omega_{i,1}} \tilde{A}_\omega^{\tilde{s}_\omega} = \left(\prod_{\omega \in \Omega_{i,1}} \tilde{A}_\omega^{2^{l_\omega}} \prod_{\omega \in \Omega_{i,2}} \tilde{A}_\omega^{\tilde{x}_\omega} \prod_{j \notin \mathcal{J}_i} A_j^{a_j^i} \right)^{\tilde{c}}. \tag{7}$$

  As $\tilde{c}$ divides s_ω for all $\omega \in \Omega_{i,1}$ we obtain the following relation:

$$\prod_{\omega \in \Omega_{i,1}} \tilde{A}_\omega^{\frac{-\tilde{s}_\omega + 2^{l_\omega}}{\tilde{c}}} \prod_{\omega \in \Omega_{i,2}} \tilde{A}_\omega^{x_\omega} \prod_{j \notin \mathcal{J}_i} A_j^{a_j^i} = 1.$$

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

The above equality implies that we have constructed d_2 witnesses for each ω-th variable $\tilde{x}_\omega = \frac{-\tilde{s}_\omega}{\tilde{c}} + 2^{l_\omega} = \frac{s^*_\omega - s_\omega}{c - c^*} + 2^{l_\omega}$, for all $\omega \in \Omega_{i,1}$. As previously, it is possible to show that these witnesses are in the right intervals, i.e. $\tilde{x}_\omega \in]2^{l_\omega} - 2^{\epsilon(\mu_\omega + k)+2}, 2^{l_\omega} + 2^{\epsilon(\mu_\omega+k)+2}[$, for all $\omega \in \Omega_{i,1}$. We can also assume in this case that $\tilde{\mathcal{P}}$ already knows those witnesses.

- It exists at least one integer $\omega \in \Omega_{i,1}$ such that $\tilde{c}$ does not divide $\tilde{s}_\omega$. Like in part (4.2), we have to prove that $\mathcal{M}$ can solve the given instance (n, Γ) of the Flexible RSA problem on G. As in the previous part, the relation (7) is true. Let $T_i = \left(\prod_{\omega \in \Omega_{i,1}} \tilde{A}_\omega^{2^{l_\omega}} \prod_{\omega \in \Omega_{i,2}} \tilde{A}_\omega^{\tilde{x}_\omega} \prod_{j \notin \mathcal{J}_i} A_j^{a_j^i} \right)$. As in part (4.2), we have, for all $\omega \in \Omega_{i,1}$, $\tilde{A}_\omega = \Gamma^{\sum_{j \in \mathcal{J}_{\omega,i}} \gamma_j}$, and we define $\theta_\omega = \sum_{j \in \mathcal{J}_{\omega,i}} \gamma_j$, for all $\omega \in \Omega_{i,1}$. With those notations, relation (7) becomes $\Gamma^{\sum_{\omega \in \Omega_{i,1}} \theta_\omega \tilde{s}_\omega} = T_i^{\tilde{c}}$. This relation has exactly the same form than relation (4). Then, it is possible to conclude similarly that $\mathcal{M}$ solves the given instance of the Flexible RSA problem on G with a non-negligible probability.

In conclusion, $\mathcal{M}$ will not be able to solve the given instance (n, Γ) of the Flexible RSA problem only if $\tilde{c}$ divides all integers $\tilde{s}_1, \ldots, \tilde{s}_r$. But in this case, it is necessary that $\tilde{\mathcal{P}}$ knows all the witnesses involved in the protocol, which is infeasible by assumption. Consequently, $\mathcal{M}$ necessarily solves the given instance (n, Γ) if it obtains as input two valid conversations from $\tilde{\mathcal{P}}$. Since the machine $\mathcal{M}$ interacts a polynomial number of times with $\tilde{\mathcal{P}}$ which runs in polynomial time, $\mathcal{M}$ solves the random instance of the Flexible RSA problem in polynomial time. Thus, under the Flexible RSA assumption, $\tilde{\mathcal{P}}$ cannot product valid conversations for the protocol of Figure 1, then the soundness of the DLRS is proved.

5 Conclusion

We have proved that many complex discrete-logarithm protocols in groups of unknown order are ZKPK under the Flexible RSA assumption. A result by Kiayias, Tsiounis and Yung appears as a particular case of our construction. It is possible to extend the work done in this paper to signature schemes using the Fiat-Shamir heuristic [15]. The security of the construction can then be proven by using the result of [21].

There is still some work to do since complex cryptographic constructions can also use ZKPK of secret values verifying some different properties not studied in this paper such as *e.g.* the proof of the "or" statement and the proof of equality of two discrete logarithms in different groups.

Acknowledgements

We are grateful to Marc Girault for his suggestions of improvement, and to anonymous referees for their valuable comments. This work has been partially financially supported by the European Commission through the IST Program under Contract IST-2002-507932 ECRYPT.

References

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. Crypto'2000, volume 1880 of LNCS, pages 255-270. Springer-Verlag, 2000.

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

2. N. Barić and B. Pfitzmann. Collision-Free Accumulators and Fail-Stop Signature Schemes Without Trees, Eurocrypt'97, volume 1233 of LNCS, pages 480-484. Springer-Verlag, 1997.
3. M. Bellare and S. Goldwasser. Verifiable Partial Key Escrow. ACM CCS'97, pages 78-91. ACM Press, 1997.
4. F. Boudot. Efficient Proofs that a Committed Number Lies in an Interval. Eurocrypt 2000, volume 1807 of LNCS, pages 431-444. Springer-Verlag, 2000.
5. J. Camenisch and A. Lysyanskaya. Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. Crypto 2002, volume 2442 of LNCS, pages 61-76. Springer-Verlag, 2002.
6. J. Camenisch and M. Michels. A Group Signature Scheme Based on an RSA-Variant. Asiacrypt'98, volume 1514 of LNCS, pages 160-174. Springer-Verlag, 1998.
7. J. Camenisch and M. Michels. Proving in Zero-Knowledge that a Number is the Product of Two Safe Primes. Eurocrypt'99, volume 1592 of LNCS, pages 107-122. Springer-Verlag, 1999.
8. S. Canard, A. Gouget, and E. Hufschmitt. A Handy Muti-Coupon System. ACNS 2006, volume 3089 of LNCS, pages 66-81. Springer-Verlag, 2006.
9. S. Canard and J. Traoré. On Fair E-cash Systems based on Group Signature Schemes. ACISP 2003, volume 2727 of LNCS, pages 237-248. Springer-Verlag, 2003.
10. A.H. Chan, Y. Frankel, and Y. Tsiounis. Easy Come - Easy Go Divisible Cash. Eurocrypt'98, volume 1403 of LNCS, pages 561-575. Springer-Verlag, 1998.
11. D. Chaum and T. Pedersen. Transferred Cash Grows in Size. Eurocrypt'92, volume 658 of LNCS, pages 390-407. Springer-Verlag, 1993.
12. R. Cramer and V. Shoup. Signature Schemes Based on the Strong RSA Assumption, ACM TISSEC 3(3), pages 161-185. ACM Press, 2000.
13. I. Damgård and E. Fujisaki, A Statistically-Hiding Integer Commitment Scheme Based on Groups with Hidden Order, Asiacrypt 2002, volume 2501 of LNCS, pages 143-159. Springer-Verlag, 2002.
14. U. Feige, A. Fiat, and A. Shamir. Zero-knowledge Proofs of Identity. Journal of Cryptology, 1(2), pages 77-94. 1988.
15. A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. Crypto'86, volume 263 of LNCS, pages 186-194. Springer-Verlag, 1987.
16. E. Fujisaki and T. Okamoto. Statistical Zero-Knowledge Protocols Solution to Identification and Signature Problems. Crypto'97, volume 1294 of LNCS, pages 16-30. Springer-Verlag, 1997.
17. R. Gennaro, T. Rabin, and H. Krawczyk. RSA-Based Undeniable Signatures, Journal of Cryptology, 13(4), pages 397-416. 2000.
18. M. Girault, G. Poupard, and J. Stern. On the Fly Authentication and Signature Schemes Based on Groups of Unknown Order. Journal of Cryptology, 19(4), pages 463-487. 2006.
19. S. Goldwasser, S. Micali, and C.W. Rackoff. The Knowledge Complexity of Interactive Proof Systems. SIAM Journal of Computing, vol. 18(1), pages 186-208. 1989.
20. A. Kiayias, Y. Tsiounis, and M. Yung. Traceable Signatures. Eurocrypt'04, volume 3027 of LNCS, pages 571-589. Springer-Verlag, 2004. Extended version at e-print cryptology archive report 2004/007, http://eprint.iacr.org/.
21. D. Pointcheval and J. Stern. Security Arguments for Digital Signatures and Blind Signatures. Journal of Cryptology, 13(3), pages 361-396. 2000.
22. C. P. Schnorr. Efficient Signature Generation for Smart Cards. Journal of Cryptology, 4(3), pages 239-252. 1991.

Appeared in W. Susilo, J. K. Liu, Y. Mu (Eds.): ProvSec 2007, LNCS 4784,pp. 122–137, 2007.

List Signature Schemes

Sébastien Canard[1], Berry Schoenmakers[2], Martijn Stam[3], and Jacques Traoré[1]

[1] France Télécom R&D, 42 rue des Coutures, F-14066 Caen, France.
[2] Technische Universiteit Eindhoven, P.O.Box 513, 5600 MB Eindhoven, The Netherlands.
[3] Dept. Computer Science, University of Bristol, Merchant Venturers Building, Woodland Road, Bristol, BS8 1UB, United Kingdom.

Abstract. A group signature scheme allows members of a group to issue signatures on behalf of the group, while hiding for each signature which group member actually issued it. A group signature scheme also involves a group manager, who is able to open any group signature by showing which group member issued it.
We introduce the concept of list signatures as a variant of group signatures which set a limit on the number of signatures each group member may issue. These limits must be enforced *without* having the group manager open signatures of honest group members—which excludes the trivial solution in which the group manager opens every signature to see whether some group members exceed their limits. Furthermore, we consider the problem of *publicly identifying* group members who exceed their limits, also *without* involving the group manager.
Keywords. Electronic voting schemes, Group signature schemes, List signature schemes.

1 Introduction

The basic functionality of a group signature scheme, as introduced by Chaum and Van Heijst [18], is to allow members of a group to issue signatures on behalf of the group, while hiding for each signature which group member actually issued it. In addition it must be possible for a group manager to open any signature issued on behalf of the group, by showing which member issued it.

The group manager is therefore in a powerful position and must be trusted not to revoke the anonymity of group members without permission. To lower the trust in the group manager it may be implemented in a distributed fashion (using threshold cryptography) such that signatures are only opened if a majority of the proxies agrees to do so.

In this paper we study methods to extend the functionality of group signatures while limiting the involvement of the group manager as much as possible. We introduce the notion of *public detection*, which in its simplest form enables any party to detect if a group member attempts to issue more than one signature on behalf of the group. Obviously, this problem can be solved by having the group manager open every signature to check whether a group member signed twice. In that case, however, all signatures by honest group members are *also* opened. Loosely speaking, we define a *list signature scheme* as a group signature scheme with public detection but without opening manager. To prevent the need to continuously rekey the scheme, we let signatures depend on a time frame and only require detection of signatures of the same user within the same time frame (and unlinkability otherwise).

We also introduce the notion of list signatures with *public identification*. This type of scheme not only allows any party to detect dishonest group members, but also for any party to identify them based on their signatures issued on behalf of the group. Hence, the absence

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

of a group manager who can identify misbehaving participants does not actually help these culprits in trying not to be identified. This is particularly useful in an on-line/off-line scenario, where signatures are verified on-line but only checked for double use later.

Group signatures are sometimes associated with applications such as electronic cash and electronic voting. In these applications, a central problem is to prevent payers from spending the same coin twice and voters from casting more than one ballot, respectively. Group signature schemes with public detection substantiate these claims by building in a mechanism to detect multiple signatures by the same group member. As explained above, we require that the scheme can be run in an *optimistic* mode: signatures of honest group members need not be opened; however, if multiple signatures by the same group member are detected, the group manager may reveal the identity by opening one of these signatures.

There are several constructs known in the literature that are related to group signatures, such as identity-escrow, anonymous credentials, concurrent signatures, ring signatures, traceable signatures, and direct anonymous attestation. We briefly discuss the last three, since they are most relevant to our new proposal of list signatures.

Ring signatures were introduced by Rivest et al. [28] as a light-weight alternative to group signatures. The important feature of a ring signature is the fact that groups are made ad-hoc without the intervention of a group manager by distilling a group public key from the public keys of the intended group members. Ring signatures as originally proposed do not possess a manager to open signatures. Recent work by Dodis et al. [21] show that ring signatures can in fact be equipped with an opening mechanism, in which case they become group signatures with vastly simplified group management. Obviously for list signatures a distinction can also be made between different types of group management.

Traceable signatures were introduced by Kiayias et al. [24]. They offer the same functionality as group signatures, but with two added possibilities called tracing and claiming. It is relatively straightforward for a signer to claim ownership of a signature as long as he knows the randomness used to create his signature (and note that, contrary to ordinary signatures, group signatures necessarily need to be randomized since they include an encryption of the identity). Kiayias et al. consider the situation where a signer can claim a signature originated from him without needing to know the randomness. Tracing a signature allows the group manager to publish a value that allows the tracing of all signatures originating from a specific signer.

Direct anonymous attestation was recently introduced by Brickell et al. [11]. It is slightly different from the schemes above, in that the user is actually split in two parts: a trusted platform module (TPM) and a host (for instance a mobile phone). If we ignore this separation, direct anonymous attestation can be seen as a group signature without the feature that a signature can be opened, but with the added functionality that signatures originating from the same user can be made linkable.

Roadmap Our first result is presented in Section 3, where we show how to efficiently proof equality of discrete logarithms in the 1-out-of-n setting. In Section 4 we give a definition of what constitutes a list signature scheme and, after introducing the basic design ideas in Section 5, we present a list signature scheme suitable for small groups (of users) in Section 6 and one for large groups in Section 7. We conclude with some remarks about the applicability

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

of list signatures to electronic voting schemes in Section 8. But we begin with introducing some notation and a brief and informal discussion of the relevant hardness assumptions.

2 Intractability Assumptions

The Strong RSA Assumption Let p' and q' be primes of equal length, such that both $p = 2p' + 1$ and $q = 2q' + 1$ are also prime. The number $n = pq$ is known as a safe RSA modulus. The group (under multiplication) of quadratic residues modulo n is denoted by $QR(n)$. It is not to hard to see that $QR(n)$ is a cyclic group of order $p'q'$.

The flexible RSA problem is defined as finding $u \in QR(n)$ and $e \in \mathbb{Z}_{>1}$ such that $u^e = y \bmod n$ when given y and n. Solving the flexible RSA problem is easy for those knowing the factorisation of n, however the strong RSA assumption states that given only y and n, the flexible RSA problem is hard to solve.

The Decision Diffie Hellman Problem Let G be a finite cyclic abelian group with generators f and g. Now suppose someone prepares two samples: in one f and g are raised to the same random power, in the other to two independently random powers (which in effect results in two random elements of G). The Decision Diffie Hellman problem is to distinguish between these two samples, given the four group elements (that is, f, g and the two powers).

The DDH problem has a nice and well-known reducibility property. If it is hard to distinguish tuples (f, g, f^x, g^x) from random tuples (f, g, f^x, g^y), then it is also hard to distinguish for example tuples (f, g, h, f^x, g^x, h^x) from random tuples (f, g, h, f^x, g^y, h^z).

If $G = QR(n)$, then G has composite order and for those knowing the group decomposition of G (i.e., knowing the factorisation of n), the DDH problem in G reduces to the DDH problem in the respective components of G. That is, the DDH in G is hard if and only if it is hard in all of its components, where the components of course are regarded as computational objects and not purely group-theoretic ones.

3 Proving Subset Relations for Sets of Discrete Logs

Suppose f, g are generators of a group G_q of prime order q for which $\log_g f$ is unknown. We present an efficient zero-knowledge proof of membership for the language consisting of tuples $(y_1, \ldots, y_n, a_1, \ldots, a_m) \in \mathsf{G}_q^{n+m}$, $1 \leq m \leq n$, satisfying

$$\{\log_f a_j \mid 1 \leq j \leq m\} \subseteq \{\log_g y_i \mid 1 \leq i \leq n\} \tag{1}$$

But before we do so, a quick reminder of the tools we use. Knowledge of a discrete logarithm can be proven using Schnorr's protocol [29]. It takes the prover a single exponentiation to perform, whereas simulating it would require a double exponentiation. Chaum and Pedersen [17] proposed an extension to proof equality (and knowledge) of discrete logarithms. A prover now needs to perform two single exponentiations and the cost of simulating are also doubled. Finally, Cramer, Damgård and Schoenmakers [19] have given a technique to proof one out of a many statements, but without revealing which statement (in fact, their technique is even more general). For the statement that is known, the costs are the same as for the prover of that statement, whereas for the other cases the costs equal that of simulation

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

(hence, if the prover happens to know more than what is required of him to proof, he can speed things up a bit).

We now return to a proof for the language we described. If $m = n = 1$ the language coincides with the language for the Chaum-Pedersen proof described above. For the general case, it is possible to directly apply the technique for proving 1-out-of-n relations using the Chaum-Pedersen proof as the basic proof to show that $\log_f a_j \in \{\log_g y_i \mid 1 \leq i \leq n\}$, for $j = 1, \ldots, m$. Indeed this is the approach followed by [13, 20] to construct proofs for similar statements. However, the total work for the proof becomes approximately $2mn$ double exponentiations.

We obtain an improved protocol by breaking up the proof in a different way. As a result we are able the reduce the total work by a factor of two, reducing it to about mn double exponentiations, which seems minimal.

The protocol is based on the following lemma.

Lemma 1. *Suppose* $\log_f g$ *is unknown. If one proves knowledge of witnesses* u_i, v_j, w_j *satisfying*

$$y_i = g^{u_i}, \textit{for } i = 1, \ldots, n \tag{2}$$

$$a_j = f^{v_j}, \textit{for } j = 1, \ldots, m \tag{3}$$

$$\exists_{i=1}^{n} a_j y_i = (fg)^{w_j}, \textit{for } j = 1, \ldots, m \tag{4}$$

then it follows that (1) is satisfied.

Proof. Consider an arbitrary j, and let i be such that $y_i a_j = (fg)^{w_j}$, hence $g^{u_i} f^{v_j} = (fg)^{w_j}$. The particular witnesses can be obtained from the knowledge extractor, but this implies that we can compute $\log_g f$ as

$$\log_g f = (u_i - w_j)/(w_j - v_j),$$

unless $v_j = w_j$, and hence $u_i = v_j = w_j$. Therefore, (1) must hold. □

Therefore, in order to prove that (1) holds, it suffices to prove that (2)–(4) holds. For (2) and (3) we need m and n Schnorr proofs, respectively. Statement (4) can be proven using the technique for proving 1-out-of-n relations, requiring the work of mn Schnorr simulations.

We will apply Lemma 1 in Section 6 to obtain an efficient list signature scheme. It can also be used to speed up Camenisch' group signature scheme [13] or multiway elections [20] by a factor of two.

4 List Signatures: Definition

We now move to a new type of signature scheme, which has as defining feature that they are linkable all along a time frame. More precisely, if a single user signs twice within the same time frame, his two signatures can be efficiently linked. Signatures of the same user in different time frames should remain unlinkable though. A stronger version allows public retrieval of the culprits identity without the intervention of a group manager. Because we regard double-spenditure as bad behaviour, we only define a minimum penalty (either detection

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

or identification). We do not require that the damage stops there, it might very well be that double-spenditure in fact results in full traceability of that user's signatures or even the ability of anyone to sign on that user's behalf. Needless to say, it is possible to pinpoint the penalty of double spenditure more precisely (note that in the real/ideal-model this is immediate [11]).

With respect to group management, we opted for a slightly less traditional approach. First of all we assume that there is already a legally binding PKI in place with which users can identify but also commit themselves. Secondly we assume that any group public key also implicitly defines the qualified group members under that key. This assumption makes it easier to define security for dynamic groups (either ad hoc or schemes allowing revocation). Even for schemes that claim a constant size group key, such as the one by Ateniese et al. [2] the public key satisfies this property if the transcripts of the Join-protocols are regarded as part of the public key. As noted by Dodis et al. [21] it is not so much the size of the theoretical group public key that matters in reality, but rather the information that is actually needed to sign and verify signatures (assuming that it has already been checked that this information is part of a valid and relevant group public key).

For our definition of list signatures we extend the existing model for group signatures (see [18, 13]) with protocols for detection and identification added and taking into account some recent developments regarding the definition of group signatures [25, 6, 5] and the discussion above.

A list signature scheme implies various entities: a group manager $\mathcal{M}$ who is responsible for the group's public key and users i who will be list members. The scheme contains protocols for the following tasks:

Key generation which produces the group's public key used to verify signatures, a private key for each group member, and a private key for the group manager. Typically key generation consists of a Setup protocol to initialize the system that will output the secret key of the manager(s) and some related public information; an interactive Join protocol between a user i and the manager after which the user becomes a member of the list, legally bound by his own signature using an existing PKI; a Revoke protocol that the group manager can use to revoke a member; and an Update protocol that group users can use to update their secret key after a change in the public key as a consequence of another user joining or leaving the qualified list of members.

Sign to produce a signature on input of a message, a time frame, the group's public key, and a group member's private key.

Verify which takes as input a message, a time frame, a signature and the group's public key and accepts iff the signature is correct for that message and time frame.

Open which is used by the group manager to prove that a group member did or did not produce a signature.

Rely to determine whether out of a list of signatures based on the same timeframe two (or, in general, k) signatures were produced by the same person (called detection procedure) and, for schemes with identification, by whom.

Note that a scheme with opening but without detection or identification is the known group signature scheme. A scheme with detection or identification but without the capability of opening is called a list signature scheme A scheme with both opening and detection could be called either a group signature scheme with detection or a list signature scheme with opening.

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

We do not reserve a name for a scheme with neither opening nor detectability possibilities because it is unclear to us how to define them formally.

We consider three security properties. For the second property we make a distinction between group signatures, list signatures and the combination of both. We only describe the properties informally and do not precise the powers of the adversary. In a formalization, the adversary will typically be modelled as a probabilistic polynomial time Turing machine run on input 1^λ that is allowed to introduce honest and corrupt users to the system, corrupt honest players, have honest players revoked, ask honest players for signatures and, if applicable, ask for signatures to be opened.

The only limitations we pose on the adversary are that it can only query one signature per honest player per timeframe and it cannot ask for opening a challenge signature (in the anonymity game). We are also cautious about allowing corruption of the group manager, especially since the group manager can be split according to functionality (e.g., a separate list manager and open manager). In the concrete schemes we will be more concrete about the level of corruption that the scheme can cope with.

Correctness An adversary cannot prevent honest group members from producing valid signatures, nor can the adversary cause the detection of the signatures of an honest group member.

Soundness An adversary can produce at most one valid signature per time frame per corrupted player without being detected, or without the identity of a corrupted group member being released. Note that signatures obtained from querying honest signers do not count as produced by the adversary. (If applicable, an adversary cannot produce a valid signature that does not open to a corrupted player.)

Anonymity Given a set of signatures over different timeframes, the adversary cannot determine whether two of them were signed by the same person.

Note that in the case of identification the second clause of correctness is superfluous. An adversary capable of causing identification of honest users already breaches anonymity. We are not aware of a similar argument for schemes with just detection.

The detectability feature of list signature schemes automatically implies that theoretically list members can disavow or claim a signature without the need to know the randomness that created it, based on the well known result that all of NP can be proven in zero-knowledge. As pointed out by Camenisch [4], this has the side-effect that a coalition encompassing all users bar one, can prove that a signature originated from the last remaining honest user. In ordinary group signature schemes, this ought not to be possible.

5 Basic Ideas for Realizing List Signature Schemes

In this section we informally describe how to adapt group signature schemes into list signature schemes. We will concentrate on detectability, since getting rid of the opening facility of group signatures, if required, is usually relatively straightforward. In the literature two important types of group signatures exist, those whose efficiency is (largely) independent of the group size

[4] Private communication, 2000

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

and those for which this is not the case, typically resulting in linear dependency. Interestingly, most large group signature schemes are based on the Strong RSA assumption coupled with (a version of) the Decisional Diffie Hellman assumption. Small group signature schemes can also be easily implemented based on groups in which the DDH problem is assumed hard.

This commonality of different schemes leads us to a generally applicable solution for turning known group signature schemes into list signature schemes. Let us first define some notation and terminology. We let G be a group of possibly unknown order in which the DDH problem is assumed to be hard, even if the group decomposition of G is known. We also assume the existence of a hash function $H : \{0,1\}^* \to G$. Furthermore, we observe that in all schemes we are aware of each user has a unique secret key x_i that lives in the exponent group (although not necessarily reduced modulo the group order).

The basic idea for performing detection is simple. Given the description of a timeframe T, the user computes and publishes $H(T)^{x_i}$, along with a NIZK-proof that the same secret key was used for the computation of $H(T)^{x_i}$ and the rest of the signature. Note that most, if not all, known constructions of group signatures already employ a NIZK-proof involving x_i during the signing state and that adapting these proofs can always be done theoretically and quite efficiently in practice (which will become clear from our examples).

It is intuitively clear that an otherwise honest signer who signs twice in the same timeframe will be caught since the values of $H(T)^{x_i}$ will collide in these two signatures. On the other hand, it is extremly unlikely that the signatures of two honest users cause detection. In fact, it will also be hard for an adversary to cause detection based on an honest user's signature and one of his own signatures, since the proof of knowledge that is part of any valid signature requires him to actually know a value x congruent to the secret key x_i of the honest user (modulo the group order). This secret is usually well protected from an adversary.

Linking signatures of the same user, but within different time frames will be difficult. Loosely speaking, the proof of knowledge will not aid an adversary in linking signatures since it is a NIZK-proof. Furthermore, the value $H(T)^{x_i}$ can be regarded to be independent of the rest of the signature if H is modelled as a random oracle. Here we use the fact that the rest of the signature is based on a group signature scheme where time frames are not defined, so H will not be queried on T in that part of the signature. Moreover, if the original group signature scheme was secure, that part of the list signature will not give the signer away. All that remains are the values $H(T)^{x_i}$ for a fixed x_i but various T. Under the random oracle model, the elements output by H are uncorrelated or, more precisely, even for an adversary who can adaptively choose the values T_j the distribution $H(T_j)_{j=1,\ldots,n}$ will be indistinguishable from the uniform distribution over G^n, provided all the T_j are different. But then the DDH assumption will guarantee that the distribution $(H(T_j), H(T_j)^x)_{j=1,\ldots,n}$ where x is randomly chosen in the exponent group is computationally indistinguishable from the uniform distribution over G^{2n}, since it is well known that the hardness of the DDH problem based on quadruples implies the hardness of the DDH problem based on any $2n$-tuple as long as n is polynomial in the security parameter.

The idea of detection using the exponentiation of a random group element to a fixed secret also appears in the context of for instance toggling schemes [31], fair E-cash systems [32, 16], direct anonymous attestation [11] and traceable signatures [24]. We note that it can be advantageous to use a different group than G to perform the detection (cf. [11]) to improve the efficiency or to ease the implementation of identification.

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

Identification A list signature scheme providing detection can also be extended to provide identification of the culprit using techniques based on secret sharing. We will assume that the group G is of large prime order q.

The easiest 2-out-of-q threshold sharing schemes for $\mathbb{Z}_q$ that exists is one based on lines. Let $s \in \mathbb{Z}_q$ be a secret, then the dealer picks a random point $r \in \mathbb{Z}_q$ and hands out shares $(X, r + sX) \in \mathbb{Z}_q^2$, i.e., points on a line. Given one point the slope s is still information theoretically hidden, but clearly two points or shares define the line and allow retrieval of the line.

A list signature scheme can be equipped with the identification functionality by requiring that, as part of his signature, the user releases a point on a line that contains his identity. Obviously a user should not be allowed to use the same point all over again, but this can easily by ensured by using a hash of the message to be signed. A problem though is that signatures in different time frames should not reveal anything, which basically requires a fresh line for each time frame. This technicality can be overcome by exploiting the linearity of the secret sharing scheme and conducting the secret sharing scheme in the exponent group $\mathbb{Z}_q$ of G, handing out shares (X, g^{r+sX}) instead. The secret s will remain the same, but the randomness r will depend both on the time frame and the identity of the user (basically by performing Diffie Hellman key agreement in some sense). Although only g^s can be reconstructed this way, identification of the user is ensured.

6 Small Groups

In this section we describe how the construction above can be used in conjunction with a group signature scheme based on the CDS-technique to perform 1-out-of-N proofs [19] and standard discrete logarithm based tools to form a list signature with identification capability. The scheme is actually of the *ad-hoc* type, meaning that the list of members belonging to a public key only need to be determined at the time of signing.

Setup The group manager picks a group G_q of prime order q and publishes (G_q, q), together with two randomly chosen generators g and h of G_q. The group G_q implicitly defines a hash-function $H : \{0,1\}^* \to \mathsf{G}_q$ and a hash-function $H' : \{0,1\}^* \to \mathbb{Z}_q$. The Decision Diffie-Hellman problem is assumed to be hard in the group G_q.
Note that the group manager does not have a secret key and is not involved in the remainder of the list signature scheme.

Join A user can join by picking a random $x \in \mathbb{Z}_q$ and publishing $y = g^x$, together with a proof of knowledge of x. The user also publishes some $z \in \mathbb{Z}_q$. A user's z has to be unique. Henceforth we will assume that user i is legally bound to his public key pair (y_i, z_i) and denote $x_i = \log_g y$.
To sign anonymously on behalf of a list of users, the group public key is computed as the concatenation of the public keys of the individual members.

Sign Let $(y_j, z_j)_{j=1,\ldots,N}$ be the group public key and let $i \in \{1, \ldots, N\}$. User i wishing to sign a message m in time frame T first computes $s = H(T, 1), t = H(T, 0)$, and $X = H'(m, T)$. It then publishes values $T_1 = t^{x_i}$ and $T_2 = s^{x_i} {z_i}^X$ together with a NIZK-proof that for some $j \in \{1, \ldots, n\}$ it holds that the user knows an $x \in \mathbb{Z}_q$ such that $y_j = g^x, T_1 = t^x$, and $T_2 z_j^{-X} = s^x$. Here the technique developed in Section 3 can be employed to save some

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

work. The user first proves knowledge of $\log_t T_1$ and than performs a 1-out-of-n proof of the equality of the discrete logaritms $\log_{gt} y_j T_1$ and $\log_s T_2 z_j^{-X}$.

Verify Verify that the public key is correct, i.e., only consists of values resulting from the Join protocol, and verify the NIZK-proof provided by the signature.

Rely If two signatures (T_1, T_2) on message m and (T_1', T_2') on message m' based on the same timeframe T satisfy $T_1 = T_1'$ compute $z' = (T_2/T_2')^{1/(X-X')}$, where $X = H'(m, T)$ and $X' = H'(m', T)$. Identify the user with $z = z'$.

6.1 Security

Our claim is that the scheme just described is secure in the ROM under the DDH assumption. Since the group manager is hardly involved in the scheme, he may also be totally corrupted, provided the key generation performed by him during Setup is guaranteed not to allow him to introduce a somehow weak group G_q.

Correctness We first remark that we only have to contend ourselves with the first clause, since we are dealing with a scheme with identification. The first clause is trivially satisfied, since an honest user will only ever commit to a public key (y, z) for which it knows $\log_g y$.

Soundness This follows from the soundness of the zero-knowledge proof used in the signing algorithm and the fact that, under anonymity, the adversary cannot know the secret key of honest users.

Anonymity This boils down to a straightforward reduction to the DDH.

6.2 Efficiency

Just to give a flavour of the efficiency of the scheme above, suppose it is implemented based on an elliptic curve group with $\lg q \approx 160$. In that case a single user's public key consists of two points on the elliptic curve, taking about 320 bits in total (using standard point compression techniques). The public key of a group of N members will be approximately $320N$ bits long. A signature takes 320 bits for publishing T_1 and T_2 plus an additional 320 bits per group member for the proof of knowledge, i.e., a grand total of $320(N + 1)$ bits for a signature.

7 Large Groups

Our proposal is based on the group signature scheme of Ateniese et al. [2]. In a nutshell, each user is issued with a triple (A, e, x) such that $A^e = a^x a_0 \bmod n$. The pair (A, e) is public and linked to the identity of the user, the value x is his secret. A signature is basically a proof of knowledge of such a pair, made non-interactively using the Fiat-Shamir heuristic (this is where the message comes in).

The scheme still needs a revocation scheme to be complete. Various revocation principles have been proposed ([3], [10], [15] and [26]), but [3] and [10] are not efficient enough and [26] is broken. There remains the article of Camenisch and Lysyanskaya [15] based on earlier work by Barić and Pfitzmann [4] and that is the one we will use in the following. Basically, a revocation manager publishes u and v such that $v = u^{\prod_i e_i}$ over all group member i. Camenisch

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

and Lysyanskaya fix a value for u and, by changing v, users are added or removed from the qualified list. To determine the qualified list, two sets are maintained, E_{add} and E_{del}. The set of qualified users is precisely those with a value $e \in E_{qual} = E_{add} \backslash E_{del}$ and the public key should at any time satisfy $v = u^{\prod_{e \in E_{qual}} e}$ (if the public key is corrupt, we will by definition assume that no user is qualified).

Clearly qualified users will be able to proof knowledge of an e-th root of v with a corresponding membership certificate simply by computing $B = u^{(\prod_i e_i)/e}$ where $e_i \in E_{qual}$. Camenisch and Lysyanskaya however do not publish u and show that in fact the scheme can be made more efficient: users only need to update their 'root' when other users are added or deleted and this update is linear in the number of modifications, but constant in the number of users that do not change status.

We claim that publishing u is beneficiary for two reasons. First of all it ensures that qualified users always have a membership certificate, even in the face of a corrupt revocation manager. Secondly it allows us to present an alternative that removes the update necessity when users are added. Normally when a user with exponent e is added, the revocation manager computes a new value v by v^e and hence all other users have to raise their secrets by e as well to keep track of the right root of v. However, the revocation manager might equally well update u by $u^{1/e}$ and computing $v^{1/e}$ for the new list member. Since this does not change v, other users do not need to change their respective B's. Note however that this trick falls outside of the standard dynamic accumulator theory used by Camenisch and Lysyanskaya [15], so we cannot carbon copy their security proof. Moreover, it requires that either $\mathcal{M}_R$ can compute roots, or it already knows a predetermined set of roots for the initial value of u, in which case $\mathcal{M}_J$ must be forced to use these predetermined e's during the Join-protocol.

We have to introduce some security parameters and refer to the original work by Ateniese et al. [2] for more details. Following Brickell et al. [11], we do provide recommendations for the parameters (in parentheses). We will need $\epsilon > 1$ and integers k (80) and l_p (1024). Moreover, λ_1 (4258), λ_2 (4096), γ_1 (4422) and γ_2 (4260) will denote lengths that define the intervals $\Lambda = (2^{\lambda_1} - 2^{\lambda_2}, 2^{\lambda_1} + 2^{\lambda_2})$ and $\Gamma = (2^{\gamma_1} - 2^{\gamma_2}, 2^{\gamma_1} + 2^{\gamma_2})$. Furthermore, we will need a collision-resistant hash function $H' : \{0,1\}^* \rightarrow \{0,1\}^k$.

7.1 Setup Protocol

Setup is run by the group manager in two different incarnations: one called $\mathcal{M}_J$ running the join protocol and one responsible for revocation $\mathcal{M}_R$. The join manager is the most important one in the sense that $\mathcal{M}_J$ should remain uncorrupted to achieve soundness, whereas $\mathcal{M}_R$ may be corrupted.

1. Manager $\mathcal{M}_J$ selects distinct primes p' and q' of size l_p, such that $p = 2p'+1$ and $q = 2q'+1$ are primes. It computes and publishes the modulus $n = pq$ together with a proof that n is a safe RSA-modulus [1]. It keeps the factorisation of n as secret key.
   We assume that publishing n implicitly defines a hash function $H : \{0,1\}^* \rightarrow QR(n)$. In the random oracle model, H' can be assumed to produce a distribution statistically close to the uniform one of generators of $QR(n)$ and that different calls to H' produce indepedent and hence uncorrelated outcomes unless the calls are on the same input. In

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

particular, nothing will 'leak' about respective discrete logarithms, not even to $\mathcal{M}_J$ who knows the factorisation of n.
2. Manager $\mathcal{M}_J$ chooses random elements a, a_0, g and h in $QR(n)$ and publishes them. It sets up (empty for now) a database DB.
3. Manager $\mathcal{M}_R$ chooses random elements u, f in $QR(n)$, sets $v = u$ and publishes u, v and f. It sets up (empty for now) E_{add} and E_{del}. For anonymity against $\mathcal{M}_R$ it is essential that f is uncorrelated to (g, h).

At the end of this protocol, we denote the list public key by $pk = (n, a, a_0, g, h, u, v, f)$ together with $\mathsf{DB}, E_{add}, E_{del}$, and denote $\mathcal{M}_J$'s private key by $sk_{\mathcal{M}} = p'$, Note that $\mathcal{M}_R$ does not have a private key specific to the group signature scheme, although it is implicitly understood to have a legally binding key outside the group signature scheme in order to sign its part of the group public key (u, v).

7.2 Join Protocol

This algorithm is based on the Join-protocol that is part of the group signature scheme by Ateniese et al. [2].

1. User i and manager $\mathcal{M}_J$ engage in a two-party protocol, at the end of which the user knows $x \in \Lambda$ and the manager knows $a^x \bmod n$. The protocol is such that the user cannot influence the choice (or distribution) of x and the manager learns nothing beyond $a^x \bmod n$ (this can be formalized [24]).
2. The manager picks a prime $e \in \Gamma$ not yet in DB and computes $A = (a^x a_0)^{1/e} \bmod n$. It sends (A, e) to the user.
3. The user checks that neither A nor e already occurs in the database, that e is a prime in Γ and that $A^e = a^x a_0 \bmod n$. If so, it returns (A, e) toghether with a legally binding signature on it incorporating (a, a_0, n).
4. Manager $\mathcal{M}_J$ adds (A, e) to the database, together with i's identity and signature.
5. Manager $\mathcal{M}_R$ checks that e is a prime in Γ that occurs only once in the database. If so, it adds e to E_{add}, sets $B = v$, sends B to the user, and updates the public key by $v = v^e \bmod n$. In our variation $\mathcal{M}_R$ leaves v alone, updates $u = u^{1/e}$ and sends $B = v^{1/e}$ to the user.

During this protocol, a new list member i will obtain from the managers a private key $sk_i = x$ and a membership certificate (A, e, B) such that $A^e = a^x a_0 \bmod n$ and $B^e = v \bmod n$. Note that the membership certicate is public: (A, e) is part of the database and B can be recomputed based on u, v and the sets E_{add} and E_{del}.

7.3 Revoke Protocol

In case of a revocation of the list member i with membership certificate (A_i, e_i, B_i), manager $\mathcal{M}_R$ has to compute a new value for v being $v^{1/e_i} \bmod n$. He then modifies adds e_i to E_{del} and publishes the new v and E_{del}.

Although it seems revocation requires an e_i'th root computation on $\mathcal{M}_R$'s behalf, involving the factorisation of n, the group manager can actually recompute v based on the fixed element

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

u and the sets E_{add} and E_{del}. Note that the workload for the group manager for revoking one group member is linear in the number of participants this way. If $\mathcal{M}_R$ does know the factorisation of n, it can obviously perform the root computation directly and independently of the number of participants.

If it is undesired to give $\mathcal{M}_R$ access to the secret key of $\mathcal{M}_J$, an efficient solution is the introduction of a second modulus n' of which only $\mathcal{M}_R$ knows the factorisation. In this case u, v, and f will be elements of $QR(n')$. Using a different group for the revocation manager only marginally complicates the proofs needed in the Sign protocol (the main tool is a proof of knowledge of a discrete logarithm with respect to different modulus, see e.g. [8, 7]).

7.4 Update Protocol

After a Registration protocol, a list member i (using E_{add}) has to replace B_i in his membership certificate by $B_i^e \bmod n$, where e is a new entry of E_{add}. However, if our variation is used where $\mathcal{M}_R$ updates u instead of v, old list members do not have to update their key when new members join.

After a Revocation protocol revoking user i, a list member $j \neq i$ (using E_{del}) has to replace the value B_j in his certificate by $B_j^b v^a \bmod n$ where a and b are such that $ae_j + be_i = 1$ and where e_i is a new entry of E_{del}.

In either case, the user j always knows a couple (e_j, B_j) such that $B_j^{e_j} = v \bmod n$.

7.5 Sign Protocol

In the original scheme of Ateniese et al., the signer first publishes $T_1 = A_i g^w \bmod n, T_2 = h^w \bmod n$, and $T_3 = g^{e_i} h^w \bmod n$, where w is a randomly chosen value, followed by a proof of knowledge of e_i, x_i, w, and $(e_i w)$ such that $T_1^e = a^x a_0 g^{(e_i w)}, T_2 = h^w, T_2^{e_i} = h^{(e_i w)}$, and $T_3 = g^{e_i} h^w$, all modulo n. The proof of knowledge includes range checks on x and e, but does not involve the primality of e. The main tool is a proof of knowledge of a discrete logarithm in a group of unknown order (see e.g. [14, 8, 7, 24]).

Our first observation is that publishing T_3 and explicitly proving knowledge of w is actually superfluous in the above scheme (or, for that matter, in a whole range of schemes derived from it). Our second observation is that T_1 and T_2 both seem necessary, even if opening of the signature based on knowledge of $\alpha = \log_g h$ is not required. Our third observation is that publishing $H(T)^x \bmod n$ will allow detectability, where T is the time frame. Finally, we remark that including Bf^w in the signature will show that the user is qualified (cf. [15]).

This leads us to the following signing algorithm for message m, time frame T and based on membership certificate (A, e, x, B).

1. The signer picks w at random modulo n, and computes

$$T_1 = Ag^w \bmod n$$
$$T_2 = h^w \bmod n$$
$$T_3 = Bf^w \bmod n$$
$$T_4 = t^x \bmod n$$

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

2. The signer proves knowledge of $e \in \Gamma, x \in \Lambda$, and existence of (ew) such that

$$T_1^e = a^x a_0 g^{(ew)} \bmod n$$
$$T_2^e = h^{(ew)} \bmod n$$
$$T_3^e = v f^{(ew)} \bmod n$$
$$T_4 = t^x \bmod n$$

where the message to be signed is hashed into the challenge.

The proof of knowledge is fairly standard and takes just over three elements, each slightly bigger than n^2.

7.6 Verify and Rely Protocols

A verifier can check the validity of a signature by simply verifying the proof of knowledge.

Moreover, everyone who has access to all signatures for a particular time frame can easily see if a user has signed twice or more by observing the value of T_4. The user cannot cheat (by using another value) because T_4 is linked with T_1 by the proof of knowledge and the private key x

7.7 Security of the Proposed List Signature Scheme

In this section, we examine the security of our list signature scheme. Once again, since we did not provide a formal model of security, we suffice with an informal argumentation why the list signature scheme just presented is secure.

Correctness The first clause of correctness is satisfied because an honest user i will only sign a pair (A, e) if he knows an x such that (A, e, x) is a valid certificate. Moreover, a user will only be classified as qualified if $e \in E_{qual}$ and $v = u^{\prod_{e \in E_{qual}} e}$, from which B satisfying $B^e = v$ can be efficiently reconstructed. Hence any honest qualified will have the knowledge to pass through the NIZK proof required for a signature (where we use completeness of this proof).

The second clause of correctness is also satisfied, since the only way an adversary can cause detection is by posting a signature for the same time frame with t^x, including a proof of knowledge of a value x' such that $x' = x \bmod pq'$ where x is the user's secret key. But if the adversary knows such an x', he can also identify signatures from that user, breaching anonymity.

Soundness We claim that the list signature scheme is sound against attacks excluding the Join manager. Note that the $\mathcal{M}_J$ can always introduce ghost members that have valid certificates but that do not appear in the database. This holds even if the revocation manager is not corrupted and uses a different group of order unknown to $\mathcal{M}_J$, since the ghost users can piggyback on the e's of honest users. Soundness is based on the following argumentation.

1. Without loss of generality we can assume that the adversary knows a value $\alpha = \log_h g$ and a value $\beta = \log_h f$, since knowledge of these values only adds strength to the adversary.

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

2. If an adversary posts a valid signature (T_1, T_2, T_3, T_4), he also proofs, by virtue of the soundness of the zero-knowledge proof in the signature, that he knows $e \in \Gamma$ and $x \in \Lambda$ for which an (ew) exists such that $T_1^e = a^x a_0 g^{(ew)}$, $T_2^e = h^{(ew)}$, and $T_3^e = v f^{(ew)}$. But then $(T_2^{-\alpha} T_1)^e = a^x a_0$ and $(T_2^{-\beta} T_3)^e = v$. Since we assume the adversary knows α and β, it can compute a valid membership certificate (A, e, x, B) based on any valid signature it issues. Note that the random oracle model is used for the soundness of the NIZK-proof.
3. By virtue of the proof of equality of x in T_4 and x in the membership certificate, each certificate can be used only once within the same timeframe without triggering detection.
4. Under the strong RSA assumption, the Join-protocol cannot be used to obtain more certificates than queried for [2, Theorem 1].
5. Under anonymity of the users (see below), the value x of honest users is unknown for the adversary, hence certificates of honest users are of no avail.
6. One cannot piggyback on the activation of an honest user because of the security (under the strong RSA assumption) of the dynamic accumulator used for list management [15].

Anonymity First of all, if the Join-protocol is correctly implemented, it will only leak $a^x \bmod n$ (some care has to be taken to shield the protocol against concurrent attacks).

In the random oracle model the proof of knowledge that is part of the signature can be proven statistically zero-knowledge (note that concurrency is not an issue, since a signature is a one-round protocol). Hence all that leaks from a signature is the quadruple (Ag^w, h^w, Bf^w, t^x). An adversary that could distinguish anything about A and/or B would brake the semantic security of ElGamal encryption, which is hard under the Decisional Diffie Hellman assumption. Hence a signature essentially only leaks t^x. Luckily the problem of distinguishing the distribution $(a, t_1, t_2, \ldots, t_l, a^x, t_1, \ldots, t_l^x)$ with a and $t_1, \ldots, t_l$ uniformly random in $QR(n)$ and x in Γ from the one of $2l + 2$ uniformly distributed elements in $QR(n)$ reduces to standard DDH.

7.8 Efficiency

The group's public key consists of the modulus n and six elements of $QR(n)$, taking in total 7168 bits. A signature involves the posting of four elements in $QR(n)$ together with a proof of knowledge of about 3 elements slightly bigger than n^2. In total a signature can be expected to take about 23250 bits. Hence for the schemes presented in this work, at around 70 participants the large group scheme becomes more efficient than the one based on small groups and the CDS-technique.

We would like to point out though that recently more efficient group signature schemes have appeared for large groups, that can also be adapted to list signature schemes. In fact, the direct anonymous attestation scheme by Brickell et al. [11] is already very close to a list signature scheme if the TPM and the host in their scheme are regarded as a single user. On the downside we would like to note that presently all large group signature schemes seem to be based on the Strong RSA assumption (and the DDH), whereas the small group schemes can be based on just a group in which the DDH is hard, allowing groups over elliptic curves. Given the current track record of attacking the respective problems [27], it is to be expected that the moduli for the large group schemes have to grow faster than the group sizes of the elliptic curves, thus the break-even point can be expected to shift preferring small group and list signature schemes.

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

8 Application to Electronic Voting

Many proposals for electronic voting schemes use blind signatures as a building block. In such schemes, each voter is issued a single blind signature for use in a particular election. During the election, the voter uses its blind signature to authenticate a vote, which is submitted via an anonymous channel. The election result is determined by collecting all submitted votes and accompanying blind signatures. The security of these voting schemes relies on the fact that blind signatures cannot be forged, accepting each blind signature at most once to prevent voters from voting multiple times. Finally, ballot secrecy is achieved since the blind signatures used to authenticate a vote do not reveal any information on the identity of the voter who submitted it.

We note that list signatures can be used as a substitute for blind signatures in such voting schemes. Instead of authenticating a vote by means of a blind signature, a list signature is used, preserving the security and privacy properties of the voting scheme. A first advantage of the use of list signatures is that there is one less interaction for the voter during the election process. Another potential advantage is that the computational complexity and storage complexity improves for the voters. In a list signature scheme, the storage for a user is $O(1)$ and the work for generating a signature is also $O(1)$ (assuming a scheme for large groups). Also, there is no practical limit on the number of signature that can be produced by a user.

When using blind signatures, a user must decide beforehand how many blind signatures it wishes to produce. There are techniques to limit the storage and work when handling batches of blind signatures. Such techniques were introduced by Chaum (at DigiCash), and later investigated by Fiat [22] and Schoenmakers [30]. These ideas can be extended to 'packing' many blind signatures, but achieving both $O(1)$ storage and $O(1)$ work may require additional assumptions.

References

1. J. Algesheimer, J. Camenisch, and V. Shoup. Efficient computation modulo a shared secret with application to the generation of shared safe-prime products. In Yung [33], pages 417–432.
2. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In M. Bellare, editor, *Advances in Cryptography—Crypto'00*, volume 1880 of *Lecture Notes in Computer Science*, pages 11–15. Springer-Verlag, 2000.
3. G. Ateniese, D. Song, and G. Tsudik. Quasi-efficient revocation of group signatures. In M. Blaze, editor, *FC'02*, volume 2357 of *Lecture Notes in Computer Science*, pages 183–197. Springer-Verlag, 2003.
4. N. Barić and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In Fumy [23], pages 480–484.
5. M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In E. Biham, editor, *Advances in Cryptography—Eurocrypt'03*, volume 2656 of *Lecture Notes in Computer Science*, pages 614–629. Springer-Verlag, 2003.
6. M. Bellare, H. Shi, and C. Zhang. Foundations of group signatures: The case of dynamic groups. Technical Report 77, IACR's ePrint Archive, 2004.
7. F. Boudot. On the soundness of Girault's scheme. Poster paper of Eurocrypt 2000. Rump Session, 2000.
8. F. Boudot and J. Traoré. Efficient publicly verifiable secret sharing schemes with fast or delayed recovery. In V. Varadharajan and Y. Mu, editors, *ICICS'99*, volume 1726 of *Lecture Notes in Computer Science*, pages 87–102. Springer-Verlag, 1999.
9. C. Boyd, editor. *Advances in Cryptography—Asiacrypt'01*, volume 2248 of *Lecture Notes in Computer Science*. Springer-Verlag, 2001.

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

10. E. Bresson and J. Stern. Efficient revocation in group signatures. In K. Kim, editor, *PKC'01*, volume 1992 of *Lecture Notes in Computer Science*, pages 190–206. Springer-Verlag, 2001.
11. E. Brickell, J. Camenisch, and L. Chen. Direct anonymous attestation, 2004.
12. C. Cachin and J. Camenisch, editors. *Advances in Cryptography—Eurocrypt'04*, volume 3027 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.
13. J. Camenisch. Efficient and generalized group signatures. In Fumy [23], pages 465–479.
14. J. Camenisch. *Group Signature Schemes and Payment Systems Based on the Discrete Logarithm Problem*. PhD thesis, ETH Zurich, 1998.
15. J. Camenisch and A. Lysyanskaya. Efficient revocatioin of anonymous group membership certificates and anonymous credentials. In Yung [33], pages 93–118.
16. S. Canard and J. Traoré. On fair e-cash systems based on group signature schemes. In R. Safavi-Naini and J. Seberry, editors, *ACISP'03*, volume 2727 of *Lecture Notes in Computer Science*, pages 237–248. Springer-Verlag, 2003.
17. D. Chaum and T. Pedersen. Wallet databases with observers. In E. Brickell, editor, *Advances in Cryptography—Crypto'92*, volume 740 of *Lecture Notes in Computer Science*, pages 89–105. Springer-Verlag, 1993.
18. D. Chaum and E. van Heyst. Group signatures. In D. W. Davies, editor, *Advances in Cryptography—Eurocrypt'91*, volume 547 of *Lecture Notes in Computer Science*, pages 257–265. Springer-Verlag, 1991.
19. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Y. Desmedt, editor, *Advances in Cryptography—Crypto'94*, volume 839 of *Lecture Notes in Computer Science*, pages 174–187. Springer-Verlag, 1994.
20. R. Cramer, R. Gennaro, and B. Schoenmakers. A secure and optimally efficient multi-authority election scheme. In Fumy [23], pages 103–118.
21. Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup. Anonymous identification in ad hoc groups. In Cachin and Camenisch [12], pages 609–626.
22. A. Fiat. Batch RSA. *Journal of Cryptology*, 10(2):75–88, 1997.
23. W. Fumy, editor. *Advances in Cryptography—Eurocrypt'97*, volume 1233 of *Lecture Notes in Computer Science*. Springer-Verlag, 1997.
24. A. Kiayias, Y. Tsiounis, and M. Yung. Traceable signatures. In Cachin and Camenisch [12], pages 571–589.
25. A. Kiayias and M. Yung. Group signatures: Provable security, efficient constructions and anonymity from trapdoor holders. Technical Report 76, IACR's ePrint Archive, 2004.
26. H. Kim, J. Lim, and D. Lee. Efficient and secure member deletion in group signature schemes. In D. Won, editor, *ICISC'00*, volume 2015 of *Lecture Notes in Computer Science*, pages 150–161. Springer-Verlag, 2000.
27. A. K. Lenstra. Unbelievable security: matching AES security using public key systems. In Boyd [9], pages 67–86.
28. R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In Boyd [9], pages 552–565.
29. C. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4:161–174, 1991.
30. B. Schoenmakers. Basic security of the ecash payment system. In B. Preneel and V. Rijmen, editors, *State of the Art in Applied Cryptography*, volume 1528 of *Lecture Notes in Computer Science*, pages 338–352. Springer-Verlag, 1997. Correct version available at http://www.win.tue.nl/ berry/papers/cosic.pdf.
31. M. Stam. Toggling schemes for electronic voting. Master's thesis, Technische Universiteit Eindhoven, 1999.
32. J. Traoré. Group Signatures and Their Relevance to Privacy-Protecting Off-Line Electronic Cash Systems. ACISP'99, volume 1587 of LNCS, pages 228-243. Springer-Verlag, 1999.
33. M. Yung, editor. *Advances in Cryptography—Crypto'02*, volume 2442 of *Lecture Notes in Computer Science*. Springer-Verlag, 2002.

Appeared in Discrete Applied Mathematics Journal, volume 154-2, pp. 189-201, 2006.

Defeating Malicious Servers in a Blind Signatures Based Voting System

Sébastien Canard, Matthieu Gaud, and Jacques Traoré

France Telecom R&D
42, rue des Coutures, BP6243
14066 Caen Cedex 4, France

Systems and Application Presentations

Abstract. In this paper, we present two failures in the blind signatures based voting system Votopia [8] which has been tested during the last World Soccer Cup. We then propose a fix which relies on *fair blind signatures*. The resulting scheme is practical, satisfies the fundamental needs of security in electronic voting, including *public verifiability*, and compares favorably with other like systems in terms of computational cost. As an illustration, our variant of Votopia has been successfully trialed during the French referendum on the European Constitution in May 2005.

1 Introduction

A blind signature scheme is a protocol allowing to get a signature from a signer such that the signer's view of the protocol cannot be linked to the resulting message-signature pair.
Blind signatures can be used in applications where anonymity of a message is required such as untraceable electronic cash or electronic voting. One of the standard electronic voting scheme using blind signatures was proposed by Fujioka, Ohta and Okamoto (FOO for short) at Auscrypt'92. Unfortunately, their scheme suffers from several major drawbacks. The main one is that all voters have to participate to the ballot counting process. This means that each voter must stay until all other voters complete the casting stage, which makes the scheme unpractical for real life. In [12], Ohkubo et al. show how to avoid this inconvenience by proposing a variant of FOO's voting scheme with a simple mix-net that allows voters to "vote and go": they need not to make any action after voting. Votopia [8] is a practical implementation of this system and has been tested during the 2002 FIFA World Cup to select the Most Valuable Players.
In this paper, we first focus on the security of Votopia. We describe two failures where the first *mix server* in the system can affect the result of the election in an unnoticeable way.
We then show how to repair Votopia [8]. The resulting scheme remains practical for large scale elections, allows voters to "vote and go" and satisfies the fundamental needs of security in electronic voting, including public verifiability (that is, anyone can check the validity of the whole voting process).
The key component that makes our voting protocol convenient for voters and publicly verifiable is a (threshold) fair blind signature scheme, a variant of a blind signature scheme that has been introduced by Stadler et al. at *Eurocrypt'95*. In this variant, the signer doesn't totally lose control of the signatures he issues. He can, with the help of a trusted authority (or a quorum of authorities), either identify from the transcript of a signing session the resulting signature (*signature tracing*) or link a message-signature pair to the corresponding signing session (*session tracing*).

Appeared in G. Di Crescenzo, A.D. Rubin (Eds.): Financial Cryptography 2006, LNCS 4107,pp. 148–153, 2006.

2 Protocol Failures in Votopia

Let us first briefly review the relevant parts of Votopia (see [8] for more details). Five basic entities are involved in this voting system: the voters ($\mathcal{V}_i$ will denote the voter i), an Electoral Authority or Admin Server $\mathcal{AS}$, the mix servers or *mix-net* $\mathcal{M}$ ($\mathcal{M}_i$ will denote the mix server i), the talliers $\mathcal{T}$ ($\mathcal{T}_j$ will denote the tallier j) and a ballot-box server or bulletin board $\mathcal{BB}$ which, as usual, is publicly readable and which every participant can write to (into his own section) but nobody can delete from. The role of these different entities will be clarified in the sequel. The system makes use of the following cryptographic primitives: a threshold encryption scheme, a digital signature scheme, a blind signature scheme and a *simple* mix-net (i.e., not *universally verifiable*). Any secure implementation of these primitives suits this system. We will therefore use generic notation to describe such primitives: $E_{\mathcal{T}}$ and $D_{\mathcal{T}}$ will denote respectively $\mathcal{T}$'s threshold encryption and decryption schemes whereas $E_{\mathcal{M}}$ will denote $\mathcal{M}$'s "encryption scheme". B and UB will denote respectively the blinding and unblinding functions of the blind signature scheme. In the sequel, we will assume that each eligible voter has a pair of keys (private and public) of an agreed signature scheme and that the corresponding public key has been certified by $\mathcal{AS}$ (in Votopia, the certificate is obtained during a registration stage). S_i (respectively $S_{\mathcal{AS}}$) will denote $\mathcal{V}_i$'s signing function (respectively $\mathcal{AS}$'s signing function), C_i the certificate of the corresponding public key and $\mathcal{V}_i$'s identifier is denoted by Id_i.

Voting Stage.

1. $\mathcal{V}_i$ selects the vote v_i of his choice and encrypts v_i with $\mathcal{T}$'s public key of the threshold encryption scheme as $x_i = E_{\mathcal{T}}(v_i)$. $\mathcal{V}_i$ blinds x_i as $e_i = B(x_i, r_i)$, where r_i is a randomly chosen blinding factor. $\mathcal{V}_i$ signs e_i as $s_i = S_i(e_i)$ and sends (Id_i, C_i, e_i, s_i) to $\mathcal{AS}$.
2. $\mathcal{AS}$ checks that the signature s_i of e_i is valid and that it comes from a registered voter who has not already submitted a blind ballot. If all these verifications are valid (the protocol is aborted otherwise), then $\mathcal{AS}$ signs e_i as $d_i = S_{\mathcal{AS}}(e_i)$ and sends d_i to $\mathcal{V}_i$. At the end of the voting phase, $\mathcal{AS}$ announces the number of voters receiving $\mathcal{AS}$'s signature, and publishes the final list $L_{\mathcal{AS}}$ of (Id_i, C_i, e_i, s_i).
3. $\mathcal{V}_i$ retrieves the desired signature y_i of ballot x_i by $y_i = UB(d_i, r_i)$. $\mathcal{V}_i$ encrypts (x_i, y_i) with the "encryption key" of the mix-net as $c_i = E_{\mathcal{M}}(x_i, y_i)$. $\mathcal{V}_i$ signs c_i as $\sigma_i = S_i(c_i)$ and sends $(Id_i, C_i, c_i, \sigma_i)$ to $\mathcal{BB}$.
4. $\mathcal{BB}$ checks the signature of the posted message and checks that Id_i appears in $L_{\mathcal{AS}}$. $\mathcal{BB}$ publishes the list $L_{\mathcal{BB}}$ of $(Id_i, C_i, c_i, \sigma_i)$.

Counting Stage.

1. $\mathcal{M}$ decrypts the list of c_i and outputs the list L of (x_i, y_i) in random order.
2. $\mathcal{T}$ checks the signature y_i of x_i. If the verification fails, $\mathcal{T}$ claims that y_i is not a valid signature on x_i by publishing (x_i, y_i). If more than t (the threshold) talliers claim about the same (x_i, y_i), the mix servers have to reveal the corresponding c_i and prove in zero-knowledge that (x_i, y_i) is the correct result of decryption of c_i (we call *back-tracing* such procedure). Each tallier checks the proofs issued by each mix server. If the checks fail, the mix server that issued a wrong proof is disqualified. If all proofs are valid, it means

Appeared in G. Di Crescenzo, A.D. Rubin (Eds.): Financial Cryptography 2006, LNCS 4107,pp. 148–153, 2006.

that the voter casts an invalid vote. Thus, the vote is excluded from further steps of the counting stage. After excluding the invalid results of mix processing, $\mathcal{T}$ (cooperatively) decrypts ballots x_i and retrieves vote v_i as $v_i = D_{\mathcal{T}}(x_i)$. $\mathcal{T}$ then publishes the result of the election.

In [8], Kim et al. emphasize on the fact that their system satisfies the "vote and go" property. In particular, this means that the voters will not have to verify their votes after the voting stage. However, we would like to underline that by "vote and go", we do not mean that the voting stage should be completed in only one step but rather that the voters should not engage in all phases of voting and especially that they should not have to participate to the counting stage (which could take place far after the casting stage). Here, we will show that if we really let the voters "vote and go" then their system doesn't satisfy the *accuracy* requirement (that is the impossibility to alter a cast ballot). More precisely, we will show that the first mix server (and only this mix) can modify the result of the election in an unnoticeable way. Indeed, since the ballots sent to $\mathcal{BB}$ are signed by the voters (see step 3 of the voting stage), this first mix can easily recognize or substitute the ballots that come from voters who are members of a political party different from its own. We distinguish two cases: the case where the number n of (encrypted) ballots sent to $\mathcal{BB}$ is smaller than the number N of voters who submitted a blind ballot to $\mathcal{AS}$ (which could correspond to the case where some voters obtained their signed ballots from $\mathcal{AS}$ and finally decided not to cast their votes) and the case where n is equal to N.

First case, we suppose that $n < N$.
Suppose that the first mix server, denoted by $\mathcal{M}_1$, has $m \leqslant N - n$ accomplices. $\mathcal{M}_1$ can ask its m accomplices to execute step 1 and step 2 of the voting stage (and consequently to obtain valid signed ballots from $\mathcal{AS}$) but not the following steps (in other words, they will not send their ballots to $\mathcal{BB}$). $\mathcal{M}_1$ can then replace m valid ballots of targeted voters by the m ballots of its accomplices. As the latter ballots are valid (they contain $\mathcal{AS}$'s signature) there will be no anomaly in the list L. The back-tracing procedure will consequently not be executed and no one will detect the subterfuge. This fraud will thus allow $\mathcal{M}_1$ and its accomplices to affect the result of the election.

Second case, we suppose that $n = N$.
In this case the attack described above cannot succeed for obvious reasons. Nevertheless, as in the previous case, we suppose that $\mathcal{M}_1$, has m ($m < N$) accomplices. $\mathcal{M}_1$ asks its accomplices to obtain valid signed ballots from $\mathcal{AS}$. But this time, the accomplices will not abstain from casting their ballots. Rather, they will send dummy ballots to $\mathcal{BB}$, while keeping the valid ballots provided by $\mathcal{AS}$ for future use. $\mathcal{M}_1$ can then replace m valid ballots of its choice by the m ballots of its accomplices. Obviously, the dummy ballots will be detected and discarded in the counting stage. (Note that the back-tracing procedure will not detect $\mathcal{M}_1$'s substitution and consequently it will not be disqualified). So $\mathcal{T}$ will decrypt the remaining ballots (after having discarded the invalid ones) and tallies the final result. Again, this subterfuge will allow $\mathcal{M}_1$ and its accomplices to modify the result of the election.

Another issue is what should be done in case of a "suicide attack": suppose that the first mix server substitutes an invalid ballot for the valid one posted by a targeted voter. The substitution will be detected after the decryption of the full list of ballots in a provisional tally,

Appeared in G. Di Crescenzo, A.D. Rubin (Eds.): Financial Cryptography 2006, LNCS 4107,pp. 148–153, 2006.

and the cheating mix-server identified. But what should be done now? Either the excluded vote is added in the final tally, in which case it will be exposed as the difference between the provisional and final tally or else the vote cannot be counted!

3 Our Electronic Voting Scheme

Our aim in this section is to repair Votopia. Before describing our main proposal, we envision several approaches that seem possible.

Possible approaches: to detect the frauds described above, we could require the active participation of the voters in the counting stage to verify that their votes were counted (i.e., that their pairs (x_i, y_i) appear in the list L). However, this would clearly contradict the "vote and go" property and it would be impractical for large scale elections to force all voters to check the result. Furthermore, although each voter can check that his or her vote has been correctly counted, no voter can be assured that all ballots have been tallied correctly. Such solution would only provide *individual verifiability* and not *public verifiability*. Moreover, it is not clear how a voter can complain to the scrutineers or officials of the election without taking the risk of compromising the privacy of his or her vote.
Another option is to have the first mix server provide a proof of correct mixing. But this doesn't solve the problem anymore. Indeed, if the first mix server colludes with the second one, as well as with malicious voters, they will still be able to change valid votes in an unnoticeable way (in a similar manner at what was done by the first mix server in section 2). This remark remains valid even if we assume that the first k mix servers (among the l servers) provide a proof of correct mixing (with $k \leqslant l - 2$). In this case a collusion of malicious voters and the $k + 1$ first servers will still be able to manipulate votes.
A radical solution to overcome such shortcomings is therefore to require that all mix servers prove that they have correctly mixed the set of encrypted ballots. In other words a solution would be to use a *universally verifiable mix-net*. But if verifiable mixes are used anyway, there is no need for blind signatures at all! So we cannot assume that the mixes are verifiable when considering the use of blind signatures. Still, the security must be ensured.
We try to solve this seemingly paradox in the next section by using a threshold fair blind signature scheme and two *simple* mix-nets (care must be taken however on the choice of these mix-nets), though *robust* against server failures (which means that when a server is unavailable, it is possible to replace it by another one). However, we would like to emphasize that in most cases, the mix servers will not have to prove that the mixing was done correctly. Our solution can then be seen as an *optimistic mix-type voting scheme* [5]. As we will see, a cheating mix server will always be detected. Therefore, if the penalties for cheating are severe this will preclude any attempt.
In the sequel, we will denote by $\mathcal{M}$ and $\mathcal{TM}$ the two sets of mix-net servers (where $\mathcal{TM}_j$ will denote the mix server j) and by $E_{\mathcal{M}}$ and $E_{\mathcal{TM}}$ their respective encryption scheme. The "private key" of $\mathcal{M}$ will be denoted by $SK_{\mathcal{M}}$ and the one of $\mathcal{TM}$ by $SK_{\mathcal{TM}}$. We will denote by $\mathcal{J}$ the revocation authorities of the threshold fair blind signature scheme ($FBSS$ for short) and by $REV_{\mathcal{J}}$ the corresponding signature tracing mechanism.

Appeared in G. Di Crescenzo, A.D. Rubin (Eds.): Financial Cryptography 2006, LNCS 4107,pp. 148–153, 2006.

We saw that the problem of Votopia comes from the fact that some votes can easily be removed or substituted by other valid ballots (by valid we mean a correctly formed ballot that has been signed by the Admin Server). We want to repair Votopia by making everybody sure that, before tallying the result of the election, the ballot box doesn't contain any fraudulent ballots. We consider that fraudulent votes can be deployed either by mix servers and/or voters and from various types of actions: adding, removing or replacing a valid ballot by another one. Owing to space limitations, we will only give a sketch of our fix.

Voting Stage. This stage is similar to the one of Votopia (see section 2). The main differences are that $x_i = E_{\mathcal{TM}}(v_i)$ instead of $x_i = E_{\mathcal{T}}(v_i)$ and that y_i is a (threshold) fair blind signature of x_i rather than a conventional blind signature. (We would also like to stress that, as usual, the voter should prove that he knows the plaintext of c_i in order to prevent *vote-duplication*). At closing time of the poll, the two lists $L_{\mathcal{BB}}$ and $L_{\mathcal{AS}}$ are compared. If Id_i appears in $L_{\mathcal{BB}}$ but not in $L_{\mathcal{AS}}$, which means that $\mathcal{V}_i$ has not requested a fair blind signature, then $(Id_i, C_i, c_i, \sigma_i)$ is removed from $L_{\mathcal{BB}}$. If a voter $\mathcal{V}_i$ has requested a fair blind signature to $\mathcal{AS}$ but has not submitted (deliberately or owing to a network failure for example) a ballot to $\mathcal{BB}$ (which means that there is an entry (Id_i, C_i, e_i, s_i) in $L_{\mathcal{AS}}$ but not a corresponding one $(Id_i, C_i, c_i, \sigma_i)$ in $L_{\mathcal{BB}}$), then the anonymity of e_i is revoked. The value $f_i = REV_{\mathcal{J}}(e_i)$ is then put in a black list RL so that everybody can recognize the message-signature pair (x_i, y_i) later. Note that by using $REV_{\mathcal{J}}$, we do not compromise the privacy of the vote v_i of $\mathcal{V}_i$: depending on the fair blind signature scheme used, the revocation authorities can at most obtain y_i but not x_i!

Counting Stage. $\mathcal{M}$ decrypts the list of c_i and outputs the list L of (x_i, y_i) in random order.
Case 1: if all pairs (x_i, y_i) are found to be correct (i.e., no pair (x_i, y_i) contains an invalid signature y_i, "belongs" to RL or is duplicated) then $SK_{\mathcal{TM}}$ is revealed (which means that all the mix servers $\mathcal{TM}_i$ have to reveal their own private keys). The ballots x_i are decrypted (using $SK_{\mathcal{TM}}$). $\mathcal{TM}$ outputs the corresponding votes v_i and then publishes the result of the election.
Case 2: otherwise for each incorrect pair (x_i, y_i), the back tracing algorithm (see section 2, step 2 of the counting phase of Votopia) is used to determine whether this anomaly comes from a mix server or a voter.
Case 2.1: if a mix server cannot prove that it has correctly decrypted and permuted such a suspicious pair (x_i, y_i), it is then disqualified. $SK_{\mathcal{M}}$ is revealed, which means that all the mix servers $\mathcal{M}_i$ have to reveal their own private keys (as we use a robust mix-net, it is then possible to retrieve the key of any malicious mix server even if the latter refuses to cooperate). The list of c_i is decrypted using $SK_{\mathcal{M}}$ and a new list L containing all the decrypted (x_i, y_i) is sent to $\mathcal{TM}$. $\mathcal{TM}$ decrypts and randomly permutes the list of x_i (but this time, the mix servers have to prove that they correctly decrypt and mix their inputs, which is costly), outputs the corresponding votes v_i in random order and publishes the result of the election (we thus solve the issue of *suicide attacks*).
Case 2.2: if no mix server cheats this means that the fraud comes from a voter. The misbehaving voter is identified (thanks to the back tracing algorithm) and the anonymity of the blind ballot e_i she submitted to $\mathcal{AS}$ is revoked. The pair (x_i, y_i) is removed from L and the revoked value $f_i = REV_{\mathcal{J}}(e_i)$ is put on the black list RL. We then redo the counting stage with the new lists L and RL until we encounter either the case 1 or the case 2.1 (in order to

Appeared in G. Di Crescenzo, A.D. Rubin (Eds.): Financial Cryptography 2006, LNCS 4107,pp. 148–153, 2006.

end the counting stage).

At the end of the protocol, the lists $L_{\mathcal{AS}}$, $L_{\mathcal{BB}}$, RL, L_0 and L (as well as the intermediate lists outputted by the mix-servers) are made public. Moreover, every step of the counting stage is public (including the back-tracing procedures). Therefore, anybody can check that only invalid ballots are discarded and that the outcome of the election is consistent with the valid cast ballots (public verifiability), provided however that all the voting entities will not collude. Indeed, if the mix-servers and the admin servers collude, they can produce as many valid ballots as they wish and substitute the ballots of legitimate voters with the ones they fraudulently produced.

We would like to underline that when all goes well (that is when no voting entity cheats), the counting stage is very fast. Moreover, the fact that a misbehaving mix server or a fraudulent voter is always detected will act as a deterrent to fraudulent behavior. Therefore, in most cases, our system will yield the result of the election very rapidly.

4 Efficiency

In this section, we analyze the efficiency of our voting system.
For our voting protocol, we used [2] as the underlying FBSS. The common parameters of this scheme are two primes p and q such that $q = (p-1)/\gamma$ for a specified integer γ where $|p| = 1024$ bits and $|q| = 160$ bits. In [2] the signature of a message m is of the form: $Sig(m) = (\zeta, \rho, \omega, \sigma_1, \sigma_2, \delta)$ where $|\zeta| = 1024$ bits and $|\rho| = |\omega| = |\sigma_1| = |\sigma_2| = |\delta| = 160$ bits (see [2] for more details). That means that the Abe and Ohkubo's fair blind signature is of size 1824 bits.
Following [8], we used, for the *simple* mix-net $\mathcal{M}$, the *hybrid mix-net* of Ohkubo and Abe [1] that combines asymmetric key exchange and symmetric encryption: it efficiently handles long plaintexts that exceed the modulus size of underlying public-key encryption as well as very short ones. Our technique applies however straightforwardly to other types of *decryption mix-nets* (where the messages are successively encrypted with each server's key).
In the sequel we will deliberately ignore the encryption of the vote v_i (recall that in our protocol the ballot x_i is defined as $x_i = E_{\mathcal{TM}}(v_i)$). The main purpose of this encryption is to provide a mean for the talliers to yield the result even when a mix server cheats. As already mentioned, a misbehaving mix server will always be detected. Consequently, there is no incentives for such mix servers to cheat. In a real implementation, we could therefore assume that they will not cheat and the encryption of the v_i's become in this case useless. This fact has been confirmed in practice: in all our trials, all the decrypted ballots were valid, so we didn't have to launch the back tracing procedure.

- During the decryption phase of our protocol, each mix computes $2 \times n$ exponentiations, where n denotes the number of ballots. We do not take into account the cost of symmetric decryptions which is negligible with respect to the cost of exponentiations.
- For each fraudulent ballot (invalid or duplicated ones), each mix server will have to compute during the back tracing process 5 exponentiations. If we denote by α the number of fraudulent ballots, the total cost, for k mix servers, during the back tracing process is

Appeared in G. Di Crescenzo, A.D. Rubin (Eds.): Financial Cryptography 2006, LNCS 4107,pp. 148–153, 2006.

$(5 \times k) \times \alpha$.
Each tallier will roughly have to compute 8 exponentiations to verify the proofs of knowledge produced by the mix servers. The total cost, for t talliers, during the back tracing process is therefore equal to $(8 \times t) \times \alpha$.
The tallying phase therefore requires $(5 \times k + 8 \times t) \times \alpha$ exponentiations.

– Following Kiayias and Yung [9], we can legitimately estimate α as 1% of n (the number of fraudulent voters). The total computational cost of our scheme is then equal to $(2.05k + 0.08t)n$. Therefore, our optimistic mix-type voting system based on fair blind signatures compares favorably, in terms of efficiency, with other mix-net based voting systems (see for example [3] or the appendix for a detailed comparison).

The table 1 below compares the real cost per server (for a total of k servers) of mixing n ballots. All estimates of the efficiency of other systems are taken from [3]. This cost is expressed as the number of exponentiations required to re-encrypt the inputs, verify correctness and decrypt the outputs. Following [3], we do not take into account the cost of additions and multiplications which is negligible with respect to the cost of exponentiations.

Scheme	Computational Cost (C.C.) for k mix servers		
	Re-enc.	Proof and Verification	Decryption
Cut and Choose ZK [11]	$2n$	$642nk$	$(2+4k)n$
Pairwise Permutations [6]	$2n$	$7n\log n(2k-1)$	$(2+4k)n$
Matrix Representation [4]	$2n$	$18n(2k-1)$	$(2+4k)n$
Polynomial Scheme [10]	$2n$	$8n(2k-1)$	$(2+4k)n$
Randomized Partial Checking [7] (Trade-off: privacy)	$2n$	$n/2(2k-1)$	$(2+4k)n$
Optimistic Mixing [5]	$6n$	$6+12k$	$(5+10k)n$
Proof of Subproduct [3] (Trade-off: correctness)	$2n$	$2\beta(2k-1)$	$(2+4k)n$
Fair Blind Signatures and hybrid mix-net [this paper]	–	$(\mathbf{5k+8t})\alpha$	$\mathbf{2kn}$

Table 1. Real cost per server and total computational cost for different electronic voting protocols including k servers and n ballots. (β is the security parameter of the scheme in [3] and e.g. $1 \leq \beta \leq 5$. See [3] for more details.) α is the number of fraudulent ballots (invalid or duplicated). We can estimate α as 1% of n.

5 Typical Implementation

A prototype of this electronic voting scheme has been implemented at France Telecom in the Java programming language. This prototype has been successfully trialed during the French referendum on the European Constitution in May 2005.
All servers (Admin Server and mix servers) were installed on IBM XSerie345 Xeon with a 2.4 GHz processor and 1 GByte of RAM and with a 2 Mbits per second network flow.
With this configuration, the blind signature phase required 2.3 seconds of running time in total (including the communication time) and the counting phase required less than 2 minutes of running time in total for nearly 1000 ballots.

Appeared in G. Di Crescenzo, A.D. Rubin (Eds.): Financial Cryptography 2006, LNCS 4107,pp. 148–153, 2006.

6 Conclusion

In this paper, we have shown some weaknesses of Votopia [8] and proposed a heuristic method, relying on fair blind signatures, to defeat our attacks. Our solution, which can be seen as an optimistic mix-type voting system based on fair blind signatures, provides, almost for free, both individual and public verifiability so that everyone can be convinced of the correctness of the voting result. In terms of efficiency, it appears that our solution, when all goes well (which is a priori always the case), is better than existing mix-net based solutions. We therefore believe that fair blind signatures could represent a more promising alternative for secure on-line voting than ordinary blind signatures and an appealing direction for future work.

References

1. M. Abe and M. Ohkubo, A Length-Invariant Hybrid Mix, *Proceedings of Asiacrypt'00*, volume 1976 of LNCS, pages 178-191, Springer-Verlag, 2000.
2. M. Abe and M. Ohkubo, Provably Secure Fair Blind Signatures with Tight Revocation, *Proceedings of Asiacrypt'01*, volume 2248 of LNCS, pages 583-601, Springer-Verlag, 2001.
3. D. Boneh and P. Golle, Almost entirely correct mixing with applications to voting, *Proceedings of ACM CCCS 2002*, pages 68-77, ACM Press, 2002.
4. J. Furukawa and K. Sako, An Efficient Scheme for Proving a Shuffle, *Proceedings of Crypto'01*, volume 2139 of LNCS, pages 368-387, Springer-Verlag, 2001.
5. P. Golle, S. Zhong, D. Boneh, M. Jakobsson and A. Juels, Optimistic Mixing for Exit-Polls, *Proceedings of Asiacrypt'02*, volume 2501 of LNCS, pages 451-465, Springer-Verlag, 2002.
6. M. Jakobsson and A. Juels, Millimix: mixing in small batches. DIMACS Technical Report 99-33.
7. M. Jakobsson, A. Juels, and R. Rivest, Making Mix-Nets Robust for Electronic Voting by Randomized Partial Checking, *Proceedings of the 11th Usenix Security Symposium*, *USENIX '02*, pages 339-353, 2002.
8. K. Kim, J. Kim, B. Lee and G. Ahn, Experimental Design of Worldwide Internet Voting System using PKI, *SSGRR2001*, L'Aquila, Italy, Aug. 6-10, 2001.
9. A. Kiayias and Moti Yung, The Vector-Ballot e-Voting Approach, *Proceedings of Financial Cryptography'04*, volume 3110 of LNCS, pages 72-89, Springer-Verlag, 2004.
10. A. Neff, A verifiable secret shuffle and its application to e-voting, *ACM-CCCS'01*, pages 116-125, ACM Press, 2001.
11. W. Ogata, K. Kurosawa, K. Sako and K. Takatani, Fault tolerant anonymous channel, *Proceedings of ICICS 1997*, volume 1334 of LNCS, pages 440-444, Springer-Verlag, 1997.
12. M. Ohkubo, F. Miura, M. Abe, A. Fujioka and T. Okamoto, An Improvement on a Practical Secret Voting Scheme, *Information Security'99*, volume 1729 of LNCS, pages 225-234, Springer-Verlag, 1999.

Appeared in G. Di Crescenzo, A.D. Rubin (Eds.): Financial Cryptography 2006, LNCS 4107,pp. 148–153, 2006.

A Client-Side Approach for Privacy-Preserving Identity Federation

Sébastien Canard, Eric Malville, and Jacques Traoré

Orange Labs
42 rue des Coutures - BP6243
14066 Caen Cedex - France

Abstract. Providing Single Sign-On (SSO) between service providers and enabling service providers to share user personal attributes are critical for both users to benefit from a seamless access to their services, and service providers to realize new business opportunities. Today, however, the users have several independent, partial identities spread over different service providers. Providing SSO and attribute sharing requires that links (federations) are established between (partial) identities. In SAML 2.0 [12], the links between identities are stored and managed at the network side by the identity providers (network-side identity federation). This model prevents the service providers from mass-correlating the partial identities they have, but the users must fully trust the identity providers. In this paper, we propose a complementary approach where the users have a full control of the links between their partial identities. It is a client-side identity federation approach, which relies on the introduction of a new cryptographic tool, called invariable partially blind signature scheme, that may be of independent interest.

1 Introduction

As Internet is a prime vehicle for business and personal interactions, more and more organizations, today, provide their customers with personalized on line services. This requires that the users have, at each service provider (SP), an identity associated to credentials for authentication and attributes (personal information) for personalization. The users have, therefore, several independent, partial identities spread over SPs. The relative independence of these partial identities enables the user to control, to some degree, the exposure of personal information complying this way with a certain level of privacy and, in particular, with the collection limitation privacy principle stating that there should be limits to the collection of personal data and that any such data should be obtained by lawful and fair means [2].

Providing Single Sign-On (SSO) between SPs and enabling SPs to share user personal attributes are critical for both users to benefit from a seamless access to their services, and SPs to realize new business opportunities. This requires that federation links are established between (partial) identities. The very nature of these links, and the way they are established and managed are critical with respect to privacy.

In what we call the network-side identity federation model in this paper, the identity federation links are managed by identity providers (IdPs) in the network and the authentication of the user is performed by the IdPs. The identity federation links are used by the IdPs in authentication claims or assertions to reference the user in the namespace of the SPs.

The identities of a user can be federated using a unique identifier shared by the IdPs and all the SPs. In the OpenID 1.x [13] approach, for instance, the users have one or more OpenID identifiers (URI) at one or more IdPs. These identifiers are presented by the users to the SPs

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

and enable the SPs to determine which IdP the authentication must be delegated to. Since the users have to remember their OpenID URI, they usually create only few OpenID identifiers. This brings to a situation where a same OpenID identifier is used to access several SPs. Should these SPs be malicious, they could collude to mass-correlate the users' personal attributes each of them have and compromise this way the collection limitation privacy principle[1]. In this model, the users must fully trust both the IdP and the SPs.

The identity federations can also be established in a more privacy-friendly way. In SAML 2.0 [12] for instance, IdPs use pseudonyms or aliases (nameID) to reference the users in the SPs' namespace and these pseudonyms differ from one SP to another. This means that the SPs cannot directly reference this user in the namespace of each other, preventing, this way, malicious SPs from colluding to mass-correlate users' identities. However, in this approach, federation aliases are still stored and managed by the IdPs in the network. This requires that the users put sufficient trust in the IdPs (and the network) to entrust them with the management of the federation links, since this information would enable the IdPs to correlate the partial identities the users have at the SPs.

The identity selector, like CardSpace [6] from Microsoft or Higgins, is a new concept that comes as a software component installed and running on the users' devices. It aims to help users better understand and represent their digital identities and, therefore, to simplify the management and the use of digital identities. The identity selectors support the usual network-side identity federation model. But they also introduce a new model, that we call a client-centric identity model, where the IdPs functions are all transferred to the user's device; Identity federations and authentications are managed and performed at the client side. This approach gives the users a full control on the federation links and enable them to keep their identities separate. But, at the same time, it requires that SPs in the network put sufficient trust in the user's device to accept authentication claims provided by the IdP hosted on the device.

In this paper we propose an intermediary model between the network-side identity federation model where all the IdP's functions are all performed in the network and the client-centric identity approach where these functions are all performed in the user's device. This model is an evolution of the SAML 2.0 identity federation model based on a new partially blind signature scheme. In this model, only the management of the identity federation links are transferred to the user's device and the actual authentication of the user is performed by IdPs in the network. This model gives the users a full control on the identity federation links while preserving the trust relationships established between SPs and IdPs that enable SPs to accept authentication claims provided by the IdPs.

The Section 2 gives an overview of the client-side identity federation approach we propose. This approach relies on blind signature schemes that have been initially introduced by Chaum [7, 8]. The existing blind signature schemes must be adapted and the Section 3 presents the adaptation we propose. Note that our new cryptographic building block, called *invariable partially blind signature*, may be of independent interest. In Section 4, we describe in details the client-side identity federation and single sign-on protocols and we finally give some implementation considerations in Section 5.

[1] The version 2.0 of OpenID enables the users to provide the URL of the IdP instead of their own OpenID URI. Therefore, disclosure of the OpenID URI is not mandatory anymore but the specifications do not give any recommendation on the way an IdP should reference a user in the namespace of the SPs.

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

2 Identity Federation: Towards a Client-Side Approach

In this section, we first present the network-side identity federation model proposed in the SAML 2.0 specifications and supported by the identity selectors, and the new client-centric identity model introduced with the identity selector concept. This section also clarifies the positioning of our client-side identity federation model with regard to these existing models and shows why these different models are complementary.

2.1 SAML 2.0: A Full Network-Side Identity Federation Model

Identity federations occur between identities at SP and identities at IdP, and is based on the use of aliases (or pseudonyms). In the network-side identity federation approach, the federation aliases are stored and managed by the IdP, and can be, like in SAML 2.0, unique on a per-identity-federation basis across all IdP and SPs (see Figure 1). Each alias corresponds to a federation link between an identity local to a SP and an identity local to an IdP (*e.g.* the pseudonym 123 is the federation link between the identities $user@SP_A$ and $user@IdP$). It is a pseudo-random value (generated either by the IdP or the SP) that has no discernible correspondence with the user's identifiers and that represents the user in the namespaces of the IdP and SP.

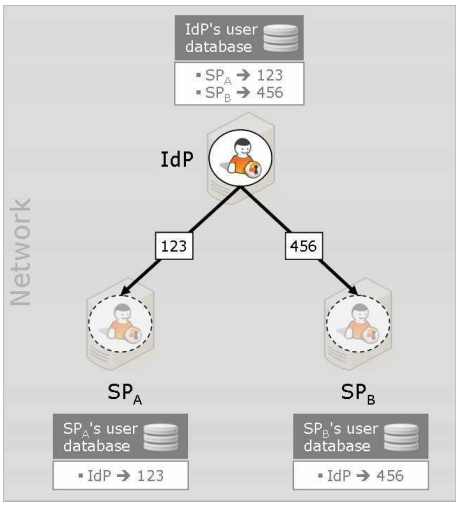

Fig. 1. Server-side identity federation

The first time a user uses a particular IdP to authenticate to a SP, the user may federate her SP's identity to her IdP's identity. This identity federation is done only once for a given user, a given IdP and a given SP. The general principle of identity federation is the following (see Figure 2).

1. The user accesses to a SP.
2. SP redirects the user to the IdP with an authentication request *AuthnRequest*. This authentication request contains in particular a request identifier, the date of the request and the identity of the SP.

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

3. IdP verifies the validity of the authentication request and authenticates the user so as to retrieve her local identity at the IdP: *user@idp*. IdP determines that the user *user@idp* has not yet federated her identity with her identity at the SP. It, therefore, generates a new federation alias *alias* and associates it to the local user identity for the SP. IdP finally generates an authentication response *AuthnResponse* and signs it. This response contains the federation alias *alias* and not the local identity *user@idp* of the user.
4. The IdP redirects the user to the SP with the authentication response *AuthnResponse*.
5. SP verifies the IdP's signature on *AuthnResponse* and extracts *alias*. Since there is no local identity corresponding to *alias*, SP asks the user whether she wants to federate his local identity with her IdP's identity. SP then authenticates the user to retrieve the local identity *user@sp* and associates it with the federation alias *alias*.
6. SP finally provides the user with the requested service. The identity *user@idp* of the user at the IdP is now federated to her identity *user@sp* at the SP with the federation alias *alias*.

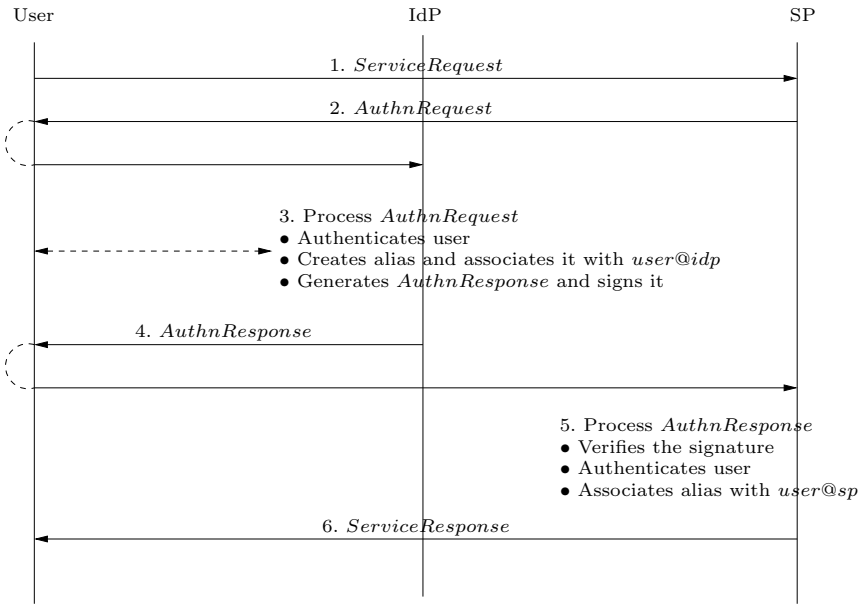

Fig. 2. SAML 2.0 identity federation

Once the identity federation is performed, the user can benefit from Single Sign On (SSO) from the IdP to the SP; the user can authenticate to the IdP and access the SP without any additional authentication. The SSO protocol relies on the same general principle as the identity federation. The only differences reside in the following two steps:

- during step 3, the IdP, after authenticating the user, retrieves the alias corresponding to *user@idp* and associates to the requested SP. As before, the IdP finally generates the authentication response *AuthnResponse* including the alias and signs it;
- during step 5, the SP does not need to authenticate the user. It first extracts the alias from the (valid) authentication request and uses this alias to retrieve the identity of the user *user@sp* from his database.

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

2.2 Identity Selector

Based on Kim Cameron's laws of Identity [10], the identity selector is a new concept materialized by a client software (*e.g.* Microsoft's CardSpace or Higgins) running on the user device, and whose goal is to simplify the management of identities by the user. The basic principle relies on the analogy with the billfold: the identity selector is a receptacle for information cards, each of them representing a user identity. The basic idea consists in symbolizing the act of authentication in order to make it more natural: in order to authenticate to a SP and provide it personal information, the user has just to present an information card.

The identity selector concept supports two identity federation models (see Figure 3). On the one hand, it supports the traditional network-side identity federation model. The card, called managed card in this case, represents an identity managed by an IdP in the network. Authentications and identity federation links are controlled and managed by this IdP and, therefore, at the network side. On the other hand, the identity selector concept introduces a new model, a client-centric identity model, in which the card (called self-issued card in this model) represents an identity created and managed locally on the user's device. The device plays the role of the IdP and the protection of this identity fully relies on the security mechanisms available and (optionally) activated locally. Authentications and identity federation links are, therefore, managed at the client side.

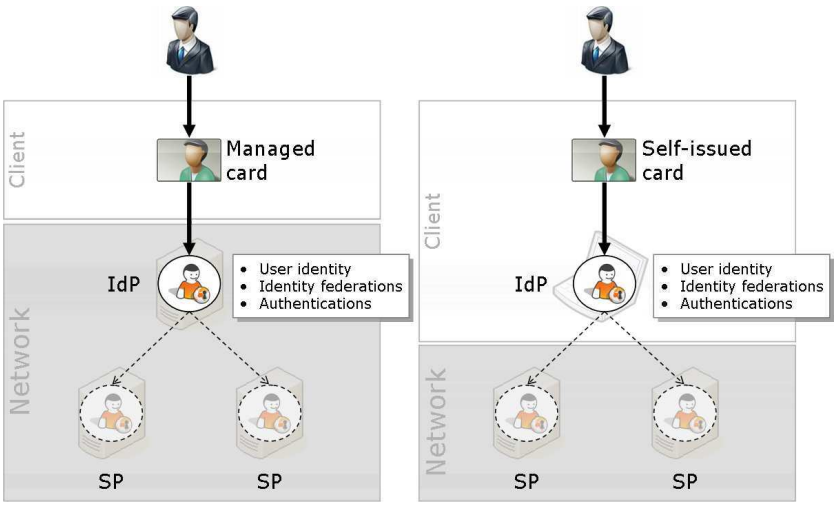

Fig. 3. Infocard model

2.3 Discussion on SAML 2.0 and the Identity Selector Concept

SAML 2.0 provides a network-side identity federation model where both the authentications and the identity federations are performed by IdPs at the network side. The Identity Provider gives, for a given user, an alias that is different from one SP to another and this alias is used by the IdPs in the authentication claims it provides to the SPs to reference the user in each SPs' namespace. SAML 2.0 provides a first level of protection of the user's privacy since there

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

is no way for two corrupted service providers to collude and mass-correlate the identities stored in their database. In most cases, this identity federation approach can be deemed acceptable from a privacy standpoint and the trust the users have in the IdPs in the network can be sufficient to entrust them with the management of the identity federation links. But in this approach, the federation aliases are stored and managed by the Identity Provider in the network. The IdP has, therefore, all the information necessary to correlate users' identities. The network-side identity federation approach is, therefore, more relevant when the partial identities are slightly independent and when the damage that the correlation of these partial identities would cause is limited. But some identities (health, government, bank...) are so different and so critical that we think the user may refuse to entrust the management of the links between these identities to the IdPs in the network.

The identity selector concept supports the network-side identity federation model and introduces a new client-centric identity model where both the authentication and identity federation functions are transferred from the network to the user's device. This new model proposed with the identity selector concept gives more control to the user since the federation aliases are controlled and managed by an identity provider residing in the user's device. It, therefore, provides an even stronger protection of the user's privacy. However, in this radical approach, authentications are performed at the client side and the SP must put sufficient trust in the device to accept the authentication claims it provides.

The client-side identity federation approach we propose (see Figure 4) is an intermediary approach between the network-side identity federation model and the client-centric identity one. It enables the user to fully control the federation aliases (preventing, this way, the IdP and the SPs to collude and mass-correlate partial user identities they have) but leave the identity management and the user authentication for IdPs in the network. As in the network-based identity federation model, the SPs can, therefore, establish trust relationships with these IdPs and accept the authentication claims they provide. This model is a complementary approach with regard to the network-side and the client-centric ones since it might be relevant in cases where the users could be reluctant to entrust the management of the identity federation links to IdPs in the network and where the SPs could refuse to trust authentication claims provided by the users' device. This could be typically the case if we want to enable authentication and attribute sharing between domains like health, government, bank... for which identities and services are deemed sensitive.

In the client-side identity federation model we propose, the IdP does not know the aliases used to actually reference the users in the namespace of the SPs. This property relies on the use of a so-called blind signature scheme. This scheme enables a requester to ask a signer to sign a message without revealing all parts of the message. The user can, therefore, ask the IdP to sign an authentication response without revealing the alias referencing the user in the SP's namespace. The assumption, here, is that the IdP does not need to know the actual alias used by the SP. The key requirement is that the identifier contained in the authentication response the IdP signs must be stable for a given user and a given SP (it must always be the same).

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

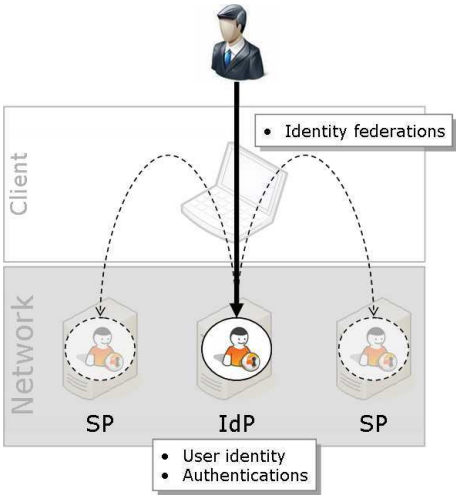

Fig. 4. Client-side federation

2.4 On the Use of Blind Signature Schemes

A blind signature scheme [7, 8] is a cryptographic protocol that allows a user to obtain a signature without giving the signer any information about the actual message. Moreover, even if the signer sees one of the documents he has signed later on, he won't be able to determine when and for whom he has signed it.

But this is not exactly what we need in our context since, as said before, the IdP does not only signs the alias but also the link between the alias, the user and the SP. Moreover, the IdP necessarily knows (i) the user's identity since the role of the IdP is to authenticate users and (ii) the requested SP since the IdP has to retrieve the right alias of the user (corresponding to this particular SP).

Consequently, the property of blind signature schemes is somewhat too restrictive. We need that the resulting signed message includes necessary information known by the signer: this is called a partially blind signature. The message to be signed is, therefore, divided into two parts: the first part M will be known by the signer whereas the second part m will be blinded (that is unknown by the signer). In our context, the blinded part of the message the IdP signs corresponds to the alias and may be some information embedded in the authentication request (such as, again, the identifier of the request) whereas the clear part will be in particular the service provider's identity. But again this is not enough in our context.

If we consider the SSO protocol, the IdP needs to sign the link between the alias, the user and the SP several times (at each new authentication request) but the (blind) alias must always be the same for a given couple (user, SP). Moreover, as mentioned before, the authentication request includes a request identifier, the date of the request and the identifier of the SP. The request identifier or the date of request can be used by the IdP and the SP to cross correlate the user's identities. Thus, this information must not be revealed to the IdP and must be different from one authentication request to another. The message must, therefore, be divided into three parts: the first one is known by the signer, the second one is blinded but needs to be invariable and the final one is a "true" blinded message. More

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

precisely, we need a new type of partially blind signature scheme, that we call invariable partially blind signature scheme. Having such building block, we can use it in our context with the message being divided into the three following parts.

1. The first one corresponds to what the IdP needs to know such as the service provider's identity.
2. The second part differs from one authentication request to another and corresponds to information that could be used by the IdP and SP for cross-correlation (the authentication request identifier or the date of the request).
3. The third part corresponds to the information that is invariable from one authentication request to another. It corresponds to the blind alias. Using invariable partially blind signature schemes, we thus obtain the new federation model shown in Figure 5.

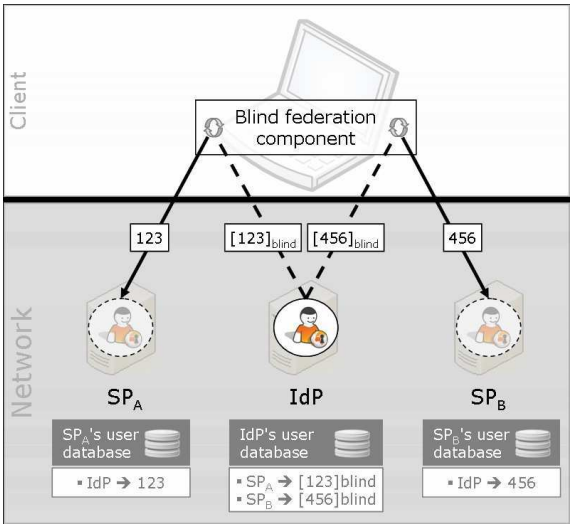

Fig. 5. Client-side identity federation model

The aim of the next section is to provide such partially blind signature scheme. In Section 4, we will finally give the way to use it in our client-side identity federation solution.

3 Invariable Partially Blind Signature Schemes

In this section, we formally introduce the notion of invariable partially blind signature scheme (IPBS for short) and next give two different ways to obtain such scheme.

3.1 Model for Invariable Partially Blind Signature Schemes

An invariable partially blind signature scheme implies three different actors: a user denoted $\mathcal{U}$, a signer $\mathcal{S}$ and a verifier $\mathcal{V}$. Each of them is implied in one or several of the following procedures:

- SETUP is an algorithm executed by *e.g.* the signer which takes on input a security parameter λ and outputs the parameters param, the signer's public key spk and the corresponding private key ssk;
- SIGN is a protocol between the user taking on input a message $\mathsf{msg} = M\|m\|m^*$, param and spk and the signer taking on input M, ssk, param and spk. The user outputs a signature σ on the message msg while the signer outputs its view viewS of the signing protocol, which includes the blinded version m_{blind} of m.
- RESIGN is a protocol between the user taking on input a message $\mathsf{msg} = M\|m\|m^*$, param, spk and optionally the blind version m_{blind} of the message m and the signer taking on input M, m_{blind}, ssk, param and spk. The user outputs a signature σ on the message $\mathsf{msg} = M\|m\|m^*$ while the signer outputs its view viewS of the re-signing protocol, which includes the blinded version m_{blind} of m.
- VERIF is an algorithm executed by the verifier which on input a message msg, a signature σ, param and spk outputs 1 if the signature σ on the message msg is valid and 0 otherwise.

We can now focus on the security properties an invariable partially blind signature scheme should verify. There are mainly four properties and if we consider useless to formally describe the correctness one which simply describe the fact that everything goes well with honest actors, it remains the three following definitions.

- **Unforgeability**: the adversary against the unforgeability property is given the signer's public key spk and param and can successfully interact ℓ_1 (resp. ℓ_2) times with the signer in the SIGN (resp. RESIGN) protocol. The adversary wins this unforgeability experiment if she successfully outputs $\ell_1 + \ell_2 + 1$ valid signatures. Thus, an invariable partially blind signature scheme is *unforgeable* if for any polynomial-time adversary, her probability of success is negligible.
- **First partial blindness**: the adversary against the first partial blindness property is given the signer's secret key ssk and param and can interact with two honest users $\mathcal{U}_0$ and $\mathcal{U}_1$. At any time of the experiment, the adversary outputs the following different messages M, $m_0\|m_0^*$ and $m_1\|m_1^*$. Then, one bit $b \in \{0,1\}$ is randomly chosen and the adversary engages in two SIGN protocols with $\mathcal{U}_0$ (resp. $\mathcal{U}_1$) taking on input $\mathsf{msg}_b = M\|m_b\|m_b^*$ (resp. $\mathsf{msg}_{\bar{b}} = M\|m_{\bar{b}}\|m_{\bar{b}}^*$), param and spk. Next, the adversary is given the two resulting signatures together with the corresponding message $(\sigma_b, \mathsf{msg}_b)$ and $(\sigma_{\bar{b}}, \mathsf{msg}_{\bar{b}})$. Finally, the adversary outputs one bit $b' \in \{0,1\}$. Thus, an invariable partially blind signature scheme is *first partially blind* if for any polynomial-time adversary, the probability that $b' = b$ differs significantly from $1/2$ is negligible.
- **Second partial blindness**: the second partial blindness is define as the first one, except that the adversary and the honest users $\mathcal{U}_0$ and $\mathcal{U}_1$ interactively play a RESIGN protocol. More precisely, the adversary is given the signer's secret key ssk and param and can interact with two honest users $\mathcal{U}_0$ and $\mathcal{U}_1$. At any time of the experiment, the adversary outputs the following different messages M, $\tilde{m}_0$, m_0^*, $\tilde{m}_1$, and m_1^* such that $m_{0_{blind}}$ and $m_{1_{blind}}$ are the blind version of the messages m_0 and m_1 that have been obtained by $\mathcal{U}_0$ and $\mathcal{U}_1$ respectively in a previously played SIGN protocol (we consider that $\mathcal{U}_0$ and $\mathcal{U}_1$ have access to m_0 and m_1 respectively). Then, one bit $b \in \{0,1\}$ is randomly chosen and the adversary engages in two RESIGN protocols with $\mathcal{U}_0$ (resp. $\mathcal{U}_1$) taking on input $\mathsf{msg}_b = M\|m_0\|m_b^*$ (resp. $\mathsf{msg}_{\bar{b}} = M\|m_1\|m_{\bar{b}}^*$), param and spk. Next, the adversary is given the two resulting signatures together with the corresponding message $(\sigma_b, \mathsf{msg}_b)$ and $(\sigma_{\bar{b}}, \mathsf{msg}_{\bar{b}})$. Finally, the

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

adversary outputs one bit $b' \in \{0,1\}$. Thus, an invariable partially blind signature scheme is *second partially blind* if for any polynomial-time adversary, the probability that $b' = b$ differs significantly from $1/2$ is negligible.

3.2 Useful Tools for Our Constructions

The basic cryptographic building blocks that will be needed for the design of our IPBS are commitment schemes, signatures of knowledge and special signature schemes (*a.k.a.* Camenisch-Lysyanskaya signature schemes).

Commitment Schemes. A commitment scheme allows a party to commit to a tuple $(x_1, x_2, \cdots, x_n)$ of (secret) values to another party. A commitment doesn't reveal any (computational) information on the tuple to the other party (hiding property) and prevents the committing party to change the values being committed to at a later stage (biding property). For example if G is a cyclic group in which the *discrete logarithm problem* is assumed to be hard and $g_1, g_2, \cdots, g_n$ are n random generators of G, then $C = \textsc{Commit}(x_1, x_2, \cdots, x_n) = g_1^{x_1} g_2^{x_2} \cdots g_n^{x_n}$ is a commitment on the tuple $(x_1, x_2, \cdots, x_n)$.

Signatures of Knowledge and Zero-Knowledge Proofs of Knowledge. In a zero-knowledge proof of knowledge a prover convinces a verifier that she knows a witness w such that a predicate P is fulfilled without releasing any further information on w. These interactive proofs can also be used non-interactively (*a.k.a signatures of knowledge*) by using the Fiat-Shamir heuristic [11]. In the sequel, we will use the following notation $NIZK[(\alpha, \beta, \cdots) : P]$ to denote a signature of knowledge proving that the prover knows a tuple $(\alpha, \beta, \cdots)$ of secret values satisfying the predicate P. In this notation, *greek letters* will denote the secret knowledge and the others letters will denote public parameters between the prover and the verifier. For example, $NIZK[\alpha : h = g^x]$ will denote a non-interactive proof of knowledge of the discrete logarithm of h in the base g (see the Schnorr authentication scheme below).

The Schnorr Authentication Scheme. The Schnorr authentication scheme has been introduced by Schnorr in [14]. It permits a prover to prove that she knows a discrete logarithm of a public key known by the verifier in a group of prime order.

Let p and q be two primes such that $q|(p-1)$. Let G be a subgroup of $\mathbb{Z}/p\mathbb{Z}$ of order q and let $g \in G$. We suppose that the prover knows the discrete logarithm x of h in base g. The authentication scheme is then described in Figure 6.

This interactive protocol can be turned non-interactively by using e.g. the Fiat-Shamir heuristic [11] that consists in first randomly choosing $w \in [0, q-1]$, computing $a = g^w$, then replacing the random choice of c by the output of a secure hash function $\mathcal{H}$ on input g and a. The final component of the non-interactive authentication is then $r = w - cx \pmod q$. The verification is done by checking that $c = \mathcal{H}(g \| g^r h^c)$. This non-interactive zero-knowledge proof is in the following denoted as $NIZK(x : h = g^x)$.

The Chaum and Pedersen Variant of the Schnorr Authentication Scheme. It exists several variants of the Schnorr authentication scheme in the literature. One is due to Brands [3] and

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

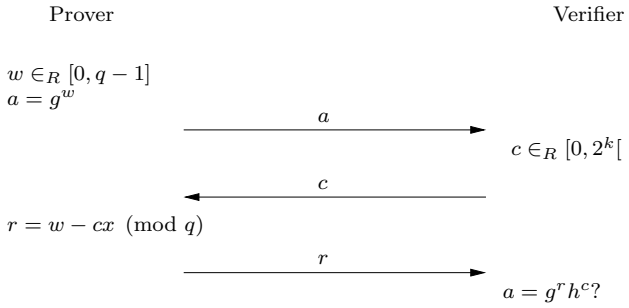

Fig. 6. Schnorr authentication scheme

permits someone to prove that she knows the representation $(x_1, \cdots, x_n)$ of a public value h in the base $(g_1, \cdots, g_n)$. Again, the non-interactive version of this proof, using the Fiat-Shamir heuristic, is denoted by $NIZK((\alpha_1, \cdots, \alpha_n : h = \prod_{i=1}^{n} g_i^{\alpha_i})$. This proof of knowledge is detailed in [3].

Another one has been proposed by Chaum and Pedersen [9]. In this latter, we consider that there exists a subgroup G of $\mathbb{Z}/p\mathbb{Z}$, where p is a prime number, of order q where q is a prime such that $q|(p-1)$. Let us also consider g and m two elements of G.

The prover owns a secret key $x \in [1, q-1]$ and computes her public keys $h = g^x$ and $z = m^x$. The aim of the Chaum and Pedersen protocol is to provide a proof of knowledge of a secret that corresponds to both the discrete logarithm of h is base g and the discrete logarithm of z in base m: this is a proof of equality of two known discrete logarithms. This is done by choosing at random $w \in [0, q-1]$, computing $a = g^w$ and $b = m^w$ and finally $r = w - cx \pmod q$ after receiving a random value c. (a, b, c, r) is valid iff $a = g^r h^c$ and $b = m^r z^c$ (see details in Figure 7).

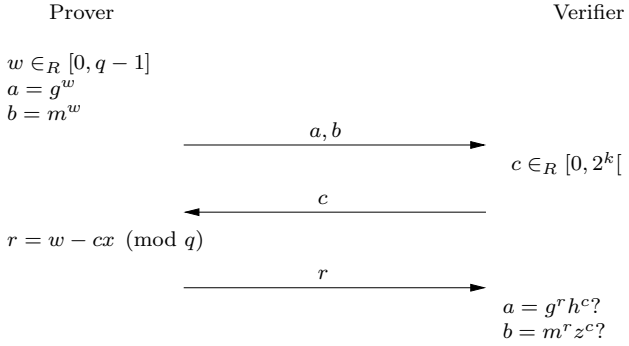

Fig. 7. Chaum-Pedersen authentication scheme

An Ad-hoc Version of the Chaum-Pedersen Scheme. It exists many ways to modify the above authentication protocols to transform it into a signature scheme. The one we describe in the following is based on the Fiat-Shamir heuristic [11]. Let p and q be two primes such that

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

$q|(p-1)$. Let G be a subgroup of $\mathbb{Z}/p\mathbb{Z}$ of order q. Let $g, g_1, g_2, g_3 \in G$. Let $\mathcal{H}$ and $\mathcal{H}_1$ be two secure hash functions.

The signer owns a private key $x \in [0, q-1]$ and publishes the corresponding public key $h = g^x$.

- Signature algorithm. Let msg be a message of the form $M\|m\|m^*$ where M and m^* are in $\{0,1\}^*$ and $m \in G$ such that the signer knows the discrete logarithm of m in base g_2. The signer acts has follow to sign msg. She first chooses at random $w \in [1, q-1]$ and computes $z = (g_1^{\mathcal{H}(M)}.m.g_3)^x$, $c = \mathcal{H}(M\|m^*\|m\|z\|g^w\|(g_1^{\mathcal{H}(M)}mg_3)^w)$, $r = w - cx \pmod q$ and finally a non-interactive proof of knowledge P of the discrete logarithm of m in base g_2 using the Schnorr protocol (see above). The signature σ on msg is thus (z, c, r, P).
- Verification algorithm. The verification of the signature (z, c, r, P) of a message $\mathsf{msg} = M\|m\|m^*$ is done by first verifying that P is valid and then by checking that

$$c = \mathcal{H}(M\|m^*\|m\|z\|g^r h^c\|(g_1^{\mathcal{H}(M)}mg_3)^r z^c).$$

This non-interactive zero-knowledge proof is denoted

$$NIZK(x : h = g^x \wedge z = m^x).$$

Camenisch-Lysyanskaya signature schemes. These special signature schemes are proposed in [4] with in addition some specific protocols.

One of this protocol allows in particular that a signature be issued on messages that are not known to the signer, but to which the signer only knows a commitment. Informally, in a protocol for signing a committed value, we have a signer with public key SPK, and the corresponding secret key SSK, and a user who queries the signer for a signature. The common input to the protocol is a commitment C on secret values $(x_1, x_2, \cdots, x_n)$ only known by the user. In the end of the protocol, the user obtains a valid Camenisch-Lysyanskaya signature $\Sigma = \text{CLSIGN}(x_1, x_2, \cdots, x_n)$ and the signer learns nothing about $(x_1, x_2, \cdots, x_n)$.

Another protocol allows to prove knowledge of a signature on a tuple of messages $(x_1, x_2, \cdots, x_n)$ without releasing any information on the corresponding signature. Each message can either be revealed to the verifier, he can know a commitment of it or have no information on it. It is for example possible to prove knowledge of a Camenisch-Lysyanskaya signature on committed values. Using our notation, such a proof would be denoted by $NIZK[(\alpha_1, \alpha_2, \cdots, \alpha_n, \beta) : C = \text{COMMIT}(\alpha_1, \alpha_2, \cdots, \alpha_n) \wedge \beta = \text{CLSIGN}(\alpha_1, \alpha_2, \cdots, \alpha_n)]$.

3.3 A First Construction Based on the Chaum-Pedersen Signature

Several blind signature schemes exist in the literature [7, 8, 5, 1] but, as explained in the above section, we need to provide a new construction, based on the model defined previously. For this purpose, we use a generic way which permits to construct a blind signature scheme by using as a primitive a discrete-log based zero-knowledge proof of knowledge (ZKPK).

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

Description. We can now introduce our invariable partially blind signature we use to improve the privacy of the identity federation of SAML 2.0 as it will be described in Section 4. For this purpose, we turn the above Chaum-Pedersen signature scheme into a blind version one where M is known by the signer and the blinded parts of the message are $m = g_2^v$ (invariable data) and m^* (variable data).

- SETUP: let p and q be two primes such that $q|(p-1)$. Let G be a subgroup of $\mathbb{Z}/p\mathbb{Z}$ of order q. Let $g, g_1, g_2, g_3 \in G$. Let $\mathcal{H}$ and $\mathcal{H}_1$ be two secure hash functions. The secret key ssk of the signer is an integer $x \in [1, q-1]$ and the corresponding public key spk is $h = g^x$.
- SIGN: in a nutshell, a user wanting to obtain a blind signature on a message $\mathsf{msg} = M\|m\|m^*$ has first to blind $m = g_2^v$ by choosing at random $s \in [1, q-1]$ and computing $m_{blind} = m.g^s$. The signer, on input m_{blind}, computes $m_0 = g_1^{\mathcal{H}_1(M)} m_{blind} g_3$ and $z_0 = m_0^x$ and proves, using the Chaum and Pedersen signature scheme, that the discrete logarithm of h in base g is the same as the discrete logarithm of z_0 in base m_0. The user finally retrieves a true Chaum-Pedersen signature $\sigma = (z, c, r, P_2)$ on the message msg as described above. The whole protocol is described more precisely in Figure 8.
- RESIGN: this procedure is played similarely to the SIGN one, except that this is not necessary to send m_{blind} anymore.
- VERIF: the verification of the validity of a blind signature on a message msg is done as described above for the Chaum and Pedersen signature scheme.

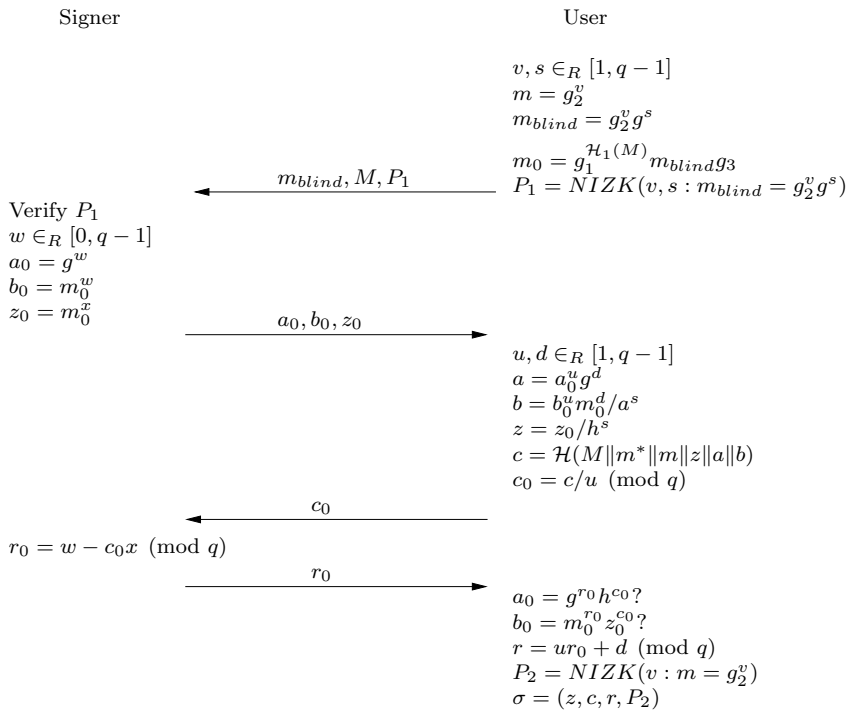

Fig. 8. The Chaum-Pedersen based IPBS scheme

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

Security Features. The security of our invariable partially blind signature scheme is given by the three following propositions. The security proofs are provided in Appendix A.

Theorem 1 (Correctness). *If both parties follow the protocol then $\sigma = (z, c, r, P_2)$ is a valid Chaum-Pedersen signature on $\mathsf{msg} = M\|m\|m^*$.*

Theorem 2 (First and second partial blindness). *For any common information M, the signer's view of the execution of the IPBS protocol is statistically independent of the messages m and m^* and the signature $\sigma = (z, c, r, P_2)$. As a consequence, our IPBS scheme verifies the first and second partial blindness properties.*

Theorem 3 (Unforgeability). *If there exists an adversary $\mathcal{A}$ against the unforgeability of our IPBS scheme, then $\mathcal{A}$ can be used to produce a signature forgery of the Chaum-Pedersen blind signature scheme.*

3.4 Another Generic Construction

Another possible way to design an invariable partially blind signature scheme is to use the so-called Camenisch-Lysyanskaya signature schemes, as introduced above. More precisely, we would have the following description for CL signature schemes based on *elliptic curves* and *bilinear maps* (see section 5 of [4]).

- SETUP: let G be a group of prime order q and g_1, g_2, g_3, g_4, g_5 five generators in G. These data constitutes the public parameters. Let $\mathcal{H}$, $\mathcal{H}_1$ and $\mathcal{H}_2$ be three secure hash functions. The signer owns a private key sk of a CL signature scheme. The corresponding public key is denoted by pk.
- SIGN: in order to obtain an invariable partially blind signature on a message $\mathsf{msg} = M\|m^*$ where $M, m^* \in \{0,1\}^*$, the user $\mathcal{U}$ will first choose three random values $a, b, r \in [1, q-1]$ and computes a commitment C' on a, b and a commitment C'' on r and $\mathcal{H}(m^*)$: $C' = g_1^a g_2^b$ and $C'' = g_3^r g_4^{\mathcal{H}(m^*)}$.[2] He will then computes two non-interactive proofs P_1 and P_2 in order to prove that he knows the committed values $P_1 = NIZK[(\alpha_1, \alpha_2) : C' = g_1^{\alpha_1} g_2^{\alpha_2}]$ and $P_2 = NIZK[(\alpha_3, \alpha_4) : C'' = g_3^{\alpha_3} g_4^{\alpha_4}]$ and send M, C', C'', P_1 and P_2 to the signer $\mathcal{S}$. The signer will verify the proofs P_1 and P_2 and if they are valid, she will choose a random value s and compute the commitment $C = C'C'' g_3^s g_5^{\mathcal{H}_2(M)}$. $\mathcal{S}$ will then use her secret key sk to compute a Camenisch-Lysyanskaya signature Σ on C (which is in fact a signature on the tuple $(a, b, r+s, \mathcal{H}_1(m^*), \mathcal{H}_2(M))$. Finally, $\mathcal{S}$ will send s and Σ to $\mathcal{U}$.
  $\mathcal{U}$ will then computes $D = g_1^a$ and a non-interactive proof P_3 that he knows a signature on these committed values:

$$P_3 = NIZK[(\alpha_1, \alpha_2, \alpha_3, \beta) : D = g_1^{\alpha_1} \wedge \\ \beta = \text{CLSIGN}(\alpha_1, \alpha_2, \alpha_3, \mathcal{H}_1(m^*), \mathcal{H}_2(M)].$$

  The pair (D, P_3) represents the invariable partially blind signature on the message $\mathsf{msg} = M\|m^*$. For our identity federation protocol D will correspond to the federation alias and C' to the blind alias. The whole protocol is described more precisely in Figure 9.

[2] The message M will be known by the signer whereas m^* should be unknown to her.

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

– ReSign: this step is simply derived from the above description and will not be detailed in this paper.
– Verif: the verification of a signature is done by checking that the non-interactive proof P_3 is correct, using standard techniques.

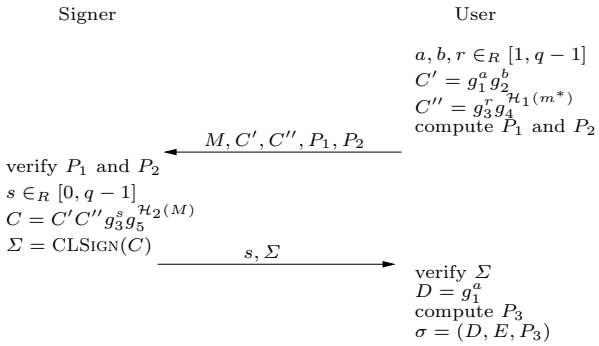

Fig. 9. The Camenisch-Lysyanskaya based IPBS scheme

4 Client-Side Identity Federation and SSO Protocols

As explained in the above sections, the aim of the identity federation approach we propose is to enable the users to control the federation links between their identities while leaving the actual user authentication to the IdPs in the network.

4.1 The Identity Federation Protocol

Again, the first time a user uses a particular IdP to authenticate to a SP, the user may federate her SP's identity to her IdP's identity. In our proposal, the reference given by the IdP to the SP with regard to a user is a blinded alias, as described in Figure 5. The global identity federation protocol is given in Figure 10 and works as follow (the IdP plays the role of the signer in the underlying invariable partially blind signature protocol whereas the SP acts as a verifier):

1. The user accesses to a SP.
2. SP redirects the user to the IdP with *AuthnRequest*. This authentication request is a standard SAML 2.0 authentication request that contains in particular a request identifier, the date of the request and the SP's identifier. The request identifier and the date of the request should be blinded (so the IdP cannot read them) since this information could be used by the IdP and the SP to establish a link between the identities.
3. IdP verifies the validity of the authentication request and authenticates the user.
4. The IdP determines that the user *user@idp* has not yet federated her identity with her identity local to the SP. The IdP and the user then engage in the invariable partially blind signature protocol given in the previous section (see Figure 8) with the following inputs.

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

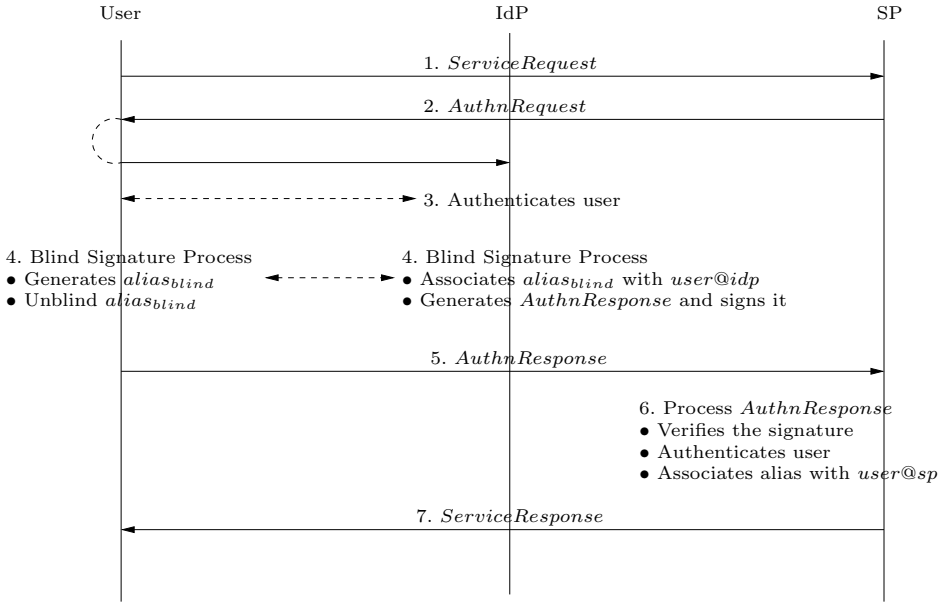

Fig. 10. Our identity federation protocol

- The message M contains the information the IdP is allowed to read including the SP's identity. As mentioned above the IdP will act as a signer in the IPBS.
- The message m^* is the information in the request/response that can be used to identify the user, such as the request identifier or the date of the request. This message is, therefore, blinded.
- The message m is the federation alias, computed as $m = g_2^v$ where v is a random value chosen by the user (see Section 5). This message is blinded.

The IdP then associates the blinded alias to the local user identity and the SP. At the end of the blind signature protocol, the user obtains an authentication response *AuthnResponse* signed by the IdP (in other words the user obtains from the IdP a valid signature on m^* and the alias m). The IdP only knows the blinded version of the alias ($alias_{blind}$) and has no way, thanks to the property of the IPBS, to compute the unblind version ($alias$).

5. The user sends the signed authentication response to the SP. This authentication response corresponds to a standard authentication response. Only the signature of the authentication response differs from the standard recommendations. The signature is composed of the following information as mentioned previously: z value, c value, r value and P_2.
6. SP verifies the IdP's signature of the *AuthnResponse* and extracts *alias*. SP then authenticates the user, retrieves the local identity $user@sp$ and associates it the federation alias *alias*.
7. Finally, SP gives the user the requested service. The identity $user@idp$ of the user at IdP is now federated to her identity $user@sp$ at SP with the federation alias *alias*.

Let us now focus on the step 4 dealing with the blind signature scheme interactions between the user and the IdP. This step can be divided as follows.

1. The IdP sends to the user a *BlindAliasRequest* to begin the blind signature protocol of Figure 8.

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

2. The user creates the alias by computing g^s, blinds it by computing m_{blind} and sends it to the IdP, together with some other values corresponding to m_0 and the proof P_1 that everything has been correctly computed. We can consider here that the part M of the global message can be either sent by the user to the IdP or directly generated by the IdP himself.
3. The next step corresponds to the exchange between the user and the signer of the blind signature protocol until the user receives r_0 from the IdP (i.e. the signer). This is also at the end of this step that the IdP associates the blinded alias m_{blind} with $user@idp$. We could moreover interpret this step as the generation and the signature by the IdP of the *AuthnResponse*. This *AuthnResponse* including the blinded alias and a blind version of the sensible information included in the authentication response is sent to the user.
4. Finally, the user processes the *AuthnResponse* by unblinding the alias and the blinded information in such a way that the signed *AuthnResponse* now includes the actual federation alias.

4.2 The Single Sign On Protocol

Once the identity federation is performed, the Single Sign On (SSO) is now possible for this user and this service provider. The new SSO protocol is again different from the identity federation one (Figure 10) since the service provider does not need to authenticate the user. The service provider uses the alias to retrieve the identity of the user. More precisely, we have the following steps.

1. The user accesses to a SP
2. SP redirects the user to the IdP with *AuthnRequest*.
3. IdP verifies the validity of the authentication request and optionally authenticates the user (authentication is not required if the user has already authenticated and if the IdP maintains an authentication session locally). IdP retrieves from his database the corresponding blinded alias for the requested SP.
4. The IdP and the user then engage in the partially blind signature protocol. More precisely, there is no need for the user to send m_{blind} to the IdP since the IdP already has it in its database. However, in the first part of the blind signature protocol of Figure 8, since the message M contains the SP's identity, the computation of m_0 and P_1 by the user are necessarily done again. At the end of the protocol, the user obtains an authentication response *AuthnResponse* signed by the IdP and containing the unblinded alias.
5. The user sends the signed authentication response to SP.
6. SP verifies the IdP's signature on *AuthnResponse*, extracts the embedded alias and retrieves the local identity $user@sp$ associated with the federation alias *alias*.
7. The SP finally sends a response to the initial service request from the user.

5 Implementation Considerations

In this section, we first give some details on security aspects that need to be studied for a practical implementation, and we give first results of the performance measurements we have performed on our own implementation.

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

5.1 Details on the Implementation

We first focus on implementation considerations by highlighting key aspects that need to be considered in order to ensure that the users' privacy is actually protected and that no security flaw is introduced.

- The blind alias together with the local identity of the user at the IdP side and the related SP should be stored in the user database of the IdP.
- In accordance with the SAML 2.0 specifications, the blind aliases should be different from one SP to another for a particular user (this prevents SPs from linking the partial identities they manage) and from one user to another.

  One possible idea is that the user and the IdP jointly generate this alias $m = g_2^v$, such that two aliases are necessarily different. For example, m could be equal to $g_2^{v_1} g_2^{v_2}$ where v_1 is secretly chosen by the user and v_2 is given by the IdP. It it moreover necessary for the user to prove, using standard ZKPK, that the sent m_{blind} is correctly formed using v_1 and v_2: $NIZK(v_1, s : m_{blind}/g_2^{v_2} = g_2^{v_1} g^s)$.
- For a particular SP and for a given user, the authentication response should always include the same alias except if one of the protagonist (user, SP or IdP) asks for a different one.
- The user should be able to verify that the alias embedded in the authentication response is not the one related to an uncorrect SP. This case may happen since this is the IdP who embeds the blinded alias in the blind signature process of the SSO protocol. Note that this verification can be easily done since the user can verify the IdP signature before sending the authentication response to the SP. If the alias is not the right one, the verification will fail since the user will not use the corresponding random values (see below).

It is important to note that, contrary to some SAML 2.0 specifications that are based on standard web browser, the approach we propose requires the installation of a blind signature module at the client side.

Moreover, some information included in the identity federation protocol needs to be used by the user during the SSO process. More precisely, the user needs to reuse the random values implied in the blinding of the alias (this is why we need a particular partially blind signature scheme, as explained in Section 2.4). Indeed, the random value s used to blind the alias g^v, chosen during the identity federation protocol between the user and the IdP is also used during all future SSO protocols. Thus, the user should have to store this value s. For this purpose, it is possible to use a single password pwd and a cryptographic function called $HMAC$. Thus this random value can e.g. be $s = HMAC(K, IdP, SP, pwd)$ such that only one password is needed to compute all necessary values. The same type of computations can also be used to pseudo-randomly choose the alias in such a way that two aliases of a same user are different from one SP to another.

One solution to the problem of random generation and storage can be to use a smart card, a dongle or a mobile phone to store the necessary values and make the computations. This can be very useful to address mobility in the client-side identity federation approach we propose.

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

5.2 First Performance Estimation

A prototype of this system has been implemented in Java, using a 32 bits Intel centrino duo processor at 1.66GHz and 2 Gbytes RAM for the client side and an Intel(R) Xeon(R) CPU 5110 at 1.60GHz for the providers.

We first consider the normal federated identity case without the blind signature process, using a 2048-bits RSA signature scheme (with CRT), and we obtain an overall running time of approximatively 300 ms.

We have implemented our first construction based on the Chaum-Pedersen signature scheme, with a 1024-bits modulus. With this configuration, the overall running time of the federation process or the SSO protocol is 1.1 s (measured from the browser). This includes the cryptographic part, the exchanges, the message formatting (in PHP), the javascript execution, etc.

The cryptographic part needs nearly 240 ms and can be divided as follows:

- client-side : 155 ms, including 45 ms for the first part of the protocol, 50 ms for the second part and optionally 60 ms to verify the obtained signature;
- IdP-side : 26 ms in total (the second step is quite immediate);
- SP-side : 60 ms to verify the signature.

6 Conclusion

In this paper, we have presented a client-side identity federation approach that complements the SAML 2.0 network-based identity federation model and the client-centric identity model introduced with the identity selector concept. In the network-side identity federation model, the users entrust the IdPs with the management of the links between their identities. In the client-side identity federation approach, the links between identities are managed at the client side by the users themselves, providing, this way, a better protection of the users' privacy. Like the network-side approach, it prevents the SPs from mass-correlating their databases. But in addition, it also prevents collusion between IdPs and SPs that would enable a mass-correlation of identity databases. Unlike the client-centric identity model, the authentications are performed by the IdPs in the network which enables the SPs to maintain their trust relationships with these IdPs and to accept the authentication claims they provide.

Our approach is based on the introduction of a new type of blind signature scheme that we call *invariable partially blind signature*. It enables the user to ask the IdP to sign authentication responses without revealing the federation alias actually used by the SP. We give an example of such blind signature scheme and apply it to our identity federation model.

Our client-side identity federation model complements the existing approaches. The users should be able to decide the most appropriate model depending on the very nature of the identities to federate and on the trust she puts on the IdPs. The network-side identity federation model is probably more relevant for partial identities that are sightly independent (an identity at Orange and an Identity at eBay for instance) and when the users put sufficient trust in the IdPs (and the network) to entrust it with the management of the federation links between these identities. On the contrary, the client-side identity federation model might be

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

chosen when the user wants to keep its identities separate. This might be the case when the identities are such that the damage that the correlation of these identities would cause is deemed important (a federation between a bank identity and a government identity for instance), and when the SPs refuse to trust authentications performed on the users' devices and to accept authentication claims provided by the IdP hosted on these devices.

However, the client-side identity federation model requires that blind signature modules are installed on the user's device. This type of cryptographic primitive has been or should be integrated in client software like Higgins or CardSpace through modules called Idemix [15][15] and U-Prove [17][18] respectively to improve user privacy and protect the user personal information. Integrating a client-side identity federation module in such client software seems to be relevant and would provide the users with an even more seamless and consistent user experience.

References

1. Masayuki Abe. A secure three-move blind signature scheme for polynomially many signatures. In EUROCRYPT '01, volume 2045 of Lecture Notes in Computer Science, pages 136-151. Springer-Verlag, 2001.
2. Liberty Alliance. http://www.projectliberty.org/.
3. Stefan A. Brands. An efficient off-line electronic cash system based on the representation problem. Technical report, Amsterdam, The Netherlands, The Netherlands, 1993.
4. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In CRYPTO 2004, volume 3152 of Lecture Notes in Computer Science, pages 56-72. Springer, 2004.
5. Jan Camenisch, Jean-Marc Piveteau, and Markus Stadler. Blind signatures based on the discrete logarithm problem. In EUROCRYPT'94, pages 428-432, 1994.
6. CardSpace. http://netfx3.com/content/windowscardspacehome.aspx.
7. David Chaum. Blind signatures for untraceable payments. In CRYPTO, pages 199-203, 1982.
8. David Chaum. Blind signature system. In CRYPTO, page 153, 1983.
9. David Chaum and Torben P. Pedersen. Wallet databases with observers. In CRYPTO'92, volume 740 of Lecture Notes in Computer Science, pages 89-105. Springer, 1992.
10. Kim Cameron (Microsoft Corporation). The laws of identity.
11. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In CRYPTO, pages 186-194, 1986.
12. E. Maler, P. Mishra, and R. Philpott. Assertions and protocol for the oasis security assertion markup language (saml). OASIS Standard, September 2003.
13. OpenID. http://openid.net/.
14. Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In CRYPTO, pages 239-252, 1989.
15. Jan Camenish and Anna Lysyanskaya - Efficient non-transferable anonymous multishow credential system with optional anonymity revocation - in EUROCRYPT 2001, vol. 2045 of LNCS, pp. 93-118, Springer Verlag 2001
16. Jan Camenisch and Els van Herreweghen - Design and implementation of the idemix anonymous credential system - Proceedings of the 9th ACM conference on Computer and communications security, 2002,ISBN 1-58113-612-9, pages 21-30 ACM, Ney York, NY, USA
17. Stefan Brands - Rethinking Public Key Infrastructures and Digital Certificates - The MIT Press, ISBN 0-262-02491-8, first edition, August 2000
18. Stefan Brands, Liesje Demuynck, Bart De Decker: A Practical System for Globally Revoking the Unlinkable Pseudonyms of Unknown Users, in ACISP 2007: vol. 4586 of LNCS, pp. 400-415, Springer Verlag 2007

A Security Analysis of our First IPBS Scheme

Let us analyze, informally speaking, the security of the invariable partially bind signature scheme described in Section 3.3 (see Figure 8).

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

A.1 Correctness

If both parties follow the protocol then $\sigma = (z, c, r, P_2)$ is a valid Chaum-Pedersen signature on $\mathsf{msg} = M\|m\|m^*$.

Let $c = \mathcal{H}(M\|m^*\|m\|z\|a\|b)$. We have to prove that

$$g^r h^c = a \text{ and } (g_1^{\mathcal{H}(M)} m g_3)^r z^c = b.$$

Thee first equality is proved as follows:

$$g^r h^c = g^{ur_0+d} h^{c_0 u} = (g^{r_0} h^{c_0})^u g^d = a_0^u g^d = a$$

For the second equality, we have:

$$\begin{aligned}(g_1^{\mathcal{H}(M)} m g_3)^r z^c &= (m_0/g^s)^r (z_0/h^s)^c = (m_0^r z_0^c)/(g^{sr} h^{sc}) \\ &= (m_0^{ur_0+d} z_0^{c_0 u})/(g^r h^c)^s \\ &= (m_0^{r_0} z_0^{c_0})^u m_0^d/(g^r h^c)^s \\ &= b_0^u m_0^d/(g^r h^c)^s = b_0^u m_0^d/a^s = b.\end{aligned}$$

Therefore the verification condition

$$c = \mathcal{H}(M\|m^*\|m\|z\|g^r h^c\|(g_1^{\mathcal{H}(M)} m g_3)^r z^c)$$

holds.

A.2 First and Second Partial Blindness

For any common information M, the signer's view of the execution of the IPBS protocol is statistically independent of the messages m and m^* and the signature $\sigma = (z, c, r, P_2)$. As a consequence, our IPBS scheme verifies the first and second partial blindness properties.

To prove that our IPBS protocol satisfies the partial blindness requirement, we have to show that for every possible view and every possible signature (on the same common information M) there exists exactly one triplet $(\tilde{u}, \tilde{d}, \tilde{s})$ of blinding factors which would result in that particular signature and view.

Given any signer's view consisting of $M, m_{blind}, a_0, b_0, z_0, c_0, r_0$ and any signature $\tilde{\sigma} = (\tilde{z}, \tilde{c}, \tilde{r}, \tilde{P_2})$ on a message $\widetilde{\mathsf{msg}} = M\|\tilde{m}\|\tilde{m}^*$ (where $\tilde{m} = g_2^{\tilde{v}}$), let $\tilde{u} = \tilde{c}/c_0 \pmod q$, $\tilde{d} = \tilde{r} - \tilde{u} r_0 \pmod q$ and $\tilde{s}$ denote the discrete logarithm of $m_{blind}/\tilde{m}$ in the base g (in other words $m_{blind} = g_2^{\tilde{v}} g^{\tilde{s}}$).

Let $a' = a_0^{\tilde{u}} g^{\tilde{d}}$ and $b' = b_0^{\tilde{u}} m_0^{\tilde{d}}/a'^{\tilde{s}}$.

It remains to show that:

$$a' = \tilde{a} = g^{\tilde{r}} h^{\tilde{c}} \tag{1}$$

$$b' = \tilde{b} = \tilde{m}_u^{\tilde{r}} \tilde{z}^{\tilde{c}} \tag{2}$$

where $\tilde{m}_u = g_1^{\mathcal{H}(M)} \tilde{m} g_3 = m_0/g^{\tilde{s}}$.

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

Let us first observe that since we assume that

$$(M, m_{blind}, a_0, b_0, z_0, c_0, r_0)$$

is a valid view and $\tilde{\sigma} = (\tilde{z}, \tilde{c}, \tilde{r}, \tilde{P}_2)$ a valid signature on $\widetilde{\mathsf{msg}} = M\|\tilde{m}\|\tilde{m}^*$, we have:

$$\begin{aligned}
a_0 &= g^{r_0} h^{c_0} \\
b_0 &= m_0^{r_0} z_0^{c_0} \\
\tilde{a} &= g^{\tilde{r}} h^{\tilde{c}} \\
\tilde{b} &= \tilde{m}_u^{\tilde{r}} \tilde{z}^{\tilde{c}} \\
\tilde{z} &= \tilde{m}_u^x = m_0^x / g^{\tilde{s}x} = z_0 / h^{\tilde{s}}
\end{aligned}$$

The last equation holds because the signer actually proves that z_0 equals m_0^x (and indirectly that $\tilde{z} = \tilde{m}_u^x$) when making a blind signature.

Let us show that the first equality (1) holds:

$$a' = a_0^{\tilde{u}} g^{\tilde{d}} = (g^{r_0} h^{c_0})^{\tilde{u}} g^{\tilde{r} - \tilde{u} r_0} = g^{r_0 \tilde{u}} h^{c_0 \tilde{u}} g^{\tilde{r} - \tilde{u} r_0} = g^{\tilde{r}} h^{\tilde{c}} = \tilde{a}.$$

For the second equality, we have:

$$\begin{aligned}
b' &= b_0^{\tilde{u}} m_0^{\tilde{d}} / a'^{\tilde{s}} = (m_0^{r_0} z_0^{c_0})^{\tilde{u}} m_0^{\tilde{d}} / (g^{\tilde{r}\tilde{s}} h^{\tilde{c}\tilde{s}}) \\
&= (m_0^{r_0 \tilde{u} + \tilde{d}} z_0^{\tilde{c}}) / (g^{\tilde{r}\tilde{s}} h^{\tilde{c}\tilde{s}}) = m_0^{\tilde{r}} z_0^{\tilde{c}} / (g^{\tilde{r}\tilde{s}} h^{\tilde{c}\tilde{s}}) \\
&= (m_0 / g^{\tilde{s}})^{\tilde{r}} (z_0 / h^{\tilde{s}})^{\tilde{c}} = \tilde{m}_u^{\tilde{r}} \tilde{z}^{\tilde{c}} = \tilde{b}.
\end{aligned}$$

Consequently $\tilde{\sigma} = (\tilde{z}, \tilde{c}, \tilde{r}, \tilde{P}_2)$ could have been produced during the execution of the IPBS protocol which led to the view consisting of $M, m_{blind}, a_0, b_0, z_0, c_0, r_0$.

This completes the proof.

A.3 Unforgeability

If there exists an adversary $\mathcal{A}$ against the unforgeability of our IPBS scheme, then $\mathcal{A}$ can be used to produce a signature forgery of the Chaum-Pedersen blind signature scheme.

The unforgeability of the signatures produced by the IPBS protocol is an immediate consequence of the security of the Chaum-Pedersen blind signature scheme [9]. Before showing that a signature forgery of the Chaum-Pedersen blind signature scheme (CPBSS for short) "reduces" to a signature forgery of our IPBS protocol, let us first recall how the CPBSS works. Let p and q be two primes such that $q|p-1$. Let G be a subgroup of $\mathbb{Z}/p\mathbb{Z}$ of order q and let $g \in G$. The secret key of the signer is an integer $x \in [1, q-1]$ and the corresponding public key is $h = g^x$. To get a blind signature on the message $\mathsf{msg} = (m^*, m)$ where $m^* \in \{0,1\}^*$ and $m \in G$, the user chooses a random $s \in [1, q-1]$ and computes $m_0 = mg^s$. Let $z_0 = m_0^x$. The signer then proves that $\log_g h = \log_{m_0} z_0$ as described in Figure 11 below.

At the end of the protocol, the user retrieves a Chaum-Pedersen signature $\sigma = (z, c, r)$ on the message $\mathsf{msg} = (m^*, m)$. The signature is considered as valid iff $c = \mathcal{H}(m^*\|m\|z\|g^r h^c\|m^r z^c)$. Note that if the execution of the protocol is successful then this implies that $z = m^x$.

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

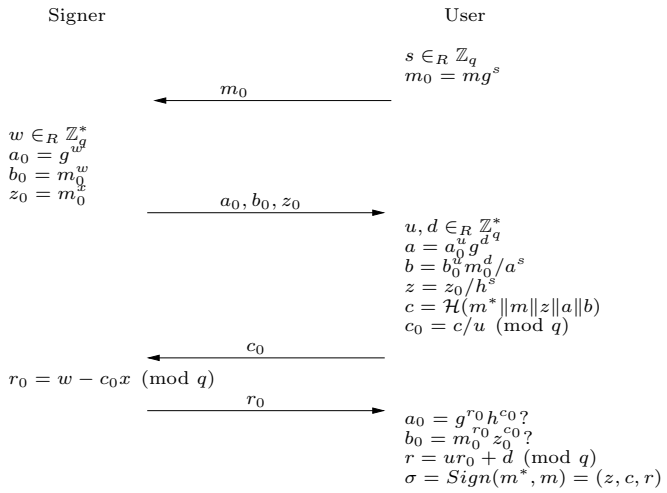

Fig. 11. Chaum-Pedersen Blind Signature Scheme

We will also need for our reduction to slightly modify the Sign protocol and the Verif algorithm of our IPBS. However, only the computation of the challenge c and consequently the verification of the validity of a signature are modified. The challenge in this variant of our IPBS is computed by the user as follows: $c = \mathcal{H}(M\|m^*\|m\|g_1^{\mathcal{H}_1(M)}mg_3\|z\|g^r h^c\|(g_1^{\mathcal{H}_1(M)}mg_3)^r z^c)$ (instead of $c = \mathcal{H}(M\|m^*\|m\|z\|g^r h^c\|(g_1^{\mathcal{H}_1(M)}mg_3)^r z^c)$). Note that when the execution of the protocol is successful we have $z = (g_1^{\mathcal{H}_1(M)}mg_3)^x$.

Suppose we have an adversary $\mathcal{A}$ who breaks the unforgeability of our IPBS with non negligible probability. We can construct an algorithm $\mathcal{B}$ using $\mathcal{A}$ as an oracle, which breaks the unforgeability of the CPBSS. $\mathcal{B}$ receives on input from its challenger $(g, h = g^x)$ the public key of the CPBSS and has oracle access to a CPBSS signature oracle (denoted by $\mathcal{O}_{CP}$ in the sequel). It also chooses a secure hash function $\mathcal{H}_1$ as well as three random generators g_1, g_2, g_3 of G and sends its public key $(g, g_1, g_2, g_3, h = g^x, \mathcal{H}_1)$ to $\mathcal{A}$.

$\mathcal{B}$ now will have to answer to the signature requests (Sign or ReSign requests) of $\mathcal{A}$. For a Sign request $\mathcal{B}$ plays as follows (the simulation is similar for ReSign requests): when $\mathcal{B}$ will receive m_{blind}, M, P_1 from $\mathcal{A}$, it will compute $m_0 = g_1^{\mathcal{H}_1(M)}mg_3$ and sends this value to $\mathcal{O}_{CP}$. $\mathcal{O}_{CP}$ will send back $a_0 = g^w, b_0 = m_0^w, z_0 = m_0^x$ to $\mathcal{B}$ that $\mathcal{B}$ will transmit to $\mathcal{A}$.

When $\mathcal{B}$ will receive the challenge c_0 from $\mathcal{A}$, $\mathcal{B}$ will transmit it to $\mathcal{O}_{CP}$. $\mathcal{O}_{CP}$ will send back a value $r_0 = w - c_0 x \pmod q$ to $\mathcal{B}$ that it will forward to $\mathcal{A}$. $\mathcal{B}$ therefore perfectly simulates the signer of the IPBS protocol.

Eventually $\mathcal{A}$ outputs $(l_1 + l_2) + 1$ valid distinct signatures (where l_1 corresponds to the number of Sign requests and l_2 to the number of ReSign requests): $(\mathsf{msg}_1 = (M_1, m_1^*, m_1), \sigma_1)$, $(\mathsf{msg}_2 = (M_2, m_2^*, m_2), \sigma_2)$, $\cdots$, $(\mathsf{msg}_{l+1} = (M_{l+1}, m_{l+1}^*, m_{l+1}), \sigma_{l+1})$ where $l = (l_1 + l_2)$.

Then $(\mathsf{MSG}_1 = (M_1\|m_1^*\|m_1\|g_1^{\mathcal{H}_1(M_1)}m_1g_3), \sigma_1)$, $(\mathsf{MSG}_2 = (M_2\|m_2^*\|m_2\|g_1^{\mathcal{H}_1(M_2)}m_2g_3), \sigma_2)$, $\cdots$, $(\mathsf{MSG}_{l+1} = (M_{l+1}\|m_{l+1}^*\|m_{l+1}\|g_1^{\mathcal{H}_1(M_{l+1})}m_{l+1}g_3), \sigma_{l+1})$ represent $l+1$ valid signatures of the CPBSS although only l signatures have been requested to $\mathcal{O}_{CP}$. $\mathcal{B}$ has therefore broken the unforgeability property of the CPBSS.

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

We have shown that if there exists an adversary $\mathcal{A}$ that can produce $l+1$ signatures after having requested only l signatures to the signer of our IPBS scheme (one-more unforgeability) then $\mathcal{A}$ can straightforwardly be used to produce a signature forgery of the Chaum-Pedersen blind signature scheme. As the later scheme is conjectured to be one-more unforgeable (see [3]) such an adversary cannot exist. A detailed security analysis of the Chaum-Pedersen blind signature scheme can be found in [3].

Appeared in Identity in the Information Society Journal, Identity Journal Limited 2009.

Divisible E-cash Systems can be Truly Anonymous*

Sébastien Canard[1] and Aline Gouget[2]

[1] France Télécom R&D, 42 rue des Coutures, F-14066 Caen, France.
[2] Gemalto, 6, rue de la Verrerie, F-92190 Meudon, France.

Abstract. This paper presents an off-line divisible e-cash scheme where a user can withdraw a divisible coin of monetary value 2^L that he can parceled and spend anonymously and unlinkably. We present the construction of a security tag that allows to protect the anonymity of honest users and to revoke anonymity only in case of cheat for protocols based on a binary tree structure without using a trusted third party. This is the first divisible e-cash scheme that provides both full unlinkability and anonymity without requiring a trusted third party.

1 Introduction

Electronic cash systems allow users to withdraw electronic coins from a bank, and then to pay a merchant using electronic coins preferably without communicating with the bank or a trusted party during the payment. Finally, the merchant deposits the spent coins to the bank.

Electronic cash provides user anonymity against both the bank and the merchant during a purchase in order to emulate the perceived anonymity of regular cash transaction. It must be impossible to link two spending protocols and a spending protocol to a withdrawal protocol.

As it is easy to duplicate electronic data, an e-cash system must prevent a user from double-spending. Ideally, the anonymity of honest users must be protected and the identity of cheaters must be recovered without using a trusted third party. An electronic payment system must also prevent a merchant from depositing the same coin twice.

To be practical, an e-cash system must be based on efficient protocols. The most critical protocol is the spending phase between the user and the merchant that must be reasonably efficient. It should also be possible to withdraw or spend several coins more efficiently than repeating several times a single withdrawal or spending protocol.

1.1 Related Works

The compact E-cash scheme [4] allows to withdraw efficiently a wallet containing 2^L coins and provides all the security properties mentioned above. One solution to improve the efficiency of the spending phase is to manage a wallet that contains coins with several monetary values as it was done in [8]; the main drawback is that the user must choose during the withdrawal protocol how many coins he wants for each monetary value.

Divisible e-cash schemes allow a user to withdraw a coin of monetary value 2^L and then to spend this coin in several times by dividing the value of the coin. The aim is to allow a user to

* This work has been partially financially supported by the European Commission through the IST Program under Contract IST-2002-507932 ECRYPT and by the French Ministry of Research RNRT Project "CRYPTO++" .

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

efficiently spend a coin of monetary value 2^ℓ, $0 \leq \ell \leq L$, (i.e. more efficiently than repeating 2^ℓ times a spending protocol). Many off-line *divisible e-cash* systems have been proposed in the literature [22, 23, 13, 14, 21, 9, 20, 19] providing part of the security properties mentioned above. The first practical divisible e-cash system was proposed by Okamoto [21] and improved by Chan *et al.* in [9]. Both schemes provide anonymity of users but not unlinkability since it is still possible to link several spends from a single divisible coin.

The first *unlinkable divisible* e-cash system that fulfills the usual properties of anonymity and unlinkability was proposed in [20] and improved in [19]. The main drawback of these two systems is that they require a trusted third party to get the identity of the user in case of double-spend detection: this is consequently what we can call a *fair* divisible e-cash system. Moreover, the unlinkability provided by [20, 19] is not strong since the merchant and the bank know which part of the withdrawn divisible coin the user is spending which is an information leak on the user.

None of the divisible e-cash schemes of the state of the art provides simultaneously strong unlinkability and truly anonymity of users.

1.2 Our Contribution

We present a strong unlinkable and anonymous divisible off-line e-cash system without trusted third party. We first provide a generic construction and next apply it to the construction of Nakanishi and Sugiyama [20]. Our system is the first that provides the user anonymity such that it is impossible for anybody to make any link between spends and withdraws. Furthermore, our construction does not require a trusted third party to revoke the anonymity of a user that has spent twice the same coin. From a theoretical point of view, the identity of the user can only be revealed when such a case happens. This is the first divisible e-cash system providing this security property.

1.3 Organization of the Paper

This paper is organized as follows. Section 2 describes the security model and requirements for a divisible e-cash system. In Section 3, we present the general principle of the construction. Section 4 is the main one: it contains the new divisible e-cash called $\mathcal{DCS}$. Finally, in Section 5, we give the security proofs of our construction.

2 Security Model

We adopt the model of divisible e-cash system without trusted third party. The three usual players are the user $\mathcal{U}$, the bank $\mathcal{B}$ and the merchant $\mathcal{M}$. The security parameter is denoted by k.

2.1 Algorithms

– `ParamKeyGen`(k)**:** a probabilistic algorithm outputting the parameters of the system $Params$ ($Params$ contains the parameter k).

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

- `BKeyGen`($Params$)**:** a probabilistic algorithm executed by $\mathcal{B}$ outputting the key pair $(sk_{\mathcal{B}}, pk_{\mathcal{B}})$.
- `KeyGen`($Params$)**:** a probabilistic algorithm executed by $\mathcal{U}$ (resp. $\mathcal{M}$) outputting $(sk_{\mathcal{U}}, pk_{\mathcal{U}})$ (resp. $(sk_{\mathcal{M}}, pk_{\mathcal{M}})$).
- `Withdraw`($\mathcal{B}(sk_{\mathcal{B}}, pk_{\mathcal{B}}, pk_{\mathcal{U}}, Params)$, $\mathcal{U}(sk_{\mathcal{U}}, pk_{\mathcal{U}}, pk_{\mathcal{B}}, Params)$)**:** an interactive protocol between $\mathcal{B}$ and $\mathcal{U}$. At the end, either $\mathcal{U}$ gets a divisible coin $\mathcal{C}$ of monetary value 2^L (L belongs to $Params$) and outputs OK, or $\mathcal{U}$ outputs $\perp$. The output of $\mathcal{B}$ is either its view $\mathcal{V}_{\mathcal{B}}^{\texttt{Withdraw}}$ of the protocol (including $pk_{\mathcal{U}}$), or $\perp$.
- `Spend`($\mathcal{U}(2^\ell, pk_{\mathcal{M}}, \mathcal{C}, Params)$, $\mathcal{M}(sk_{\mathcal{M}}, pk_{\mathcal{B}}, Params)$)**:** an interactive protocol between $\mathcal{U}$ and $\mathcal{M}$. At the end, either $\mathcal{M}$ obtains a master serial number S and a proof of validity Π and outputs (S, Π) or $\mathcal{M}$ outputs $\perp$. Either $\mathcal{U}$ updates $\mathcal{C}$ by saving the part of the divisible coin he spent (i.e. the value S) and outputs OK, or $\mathcal{U}$ outputs $\perp$.
- `Deposit` ($\mathcal{M}((S, \Pi), sk_{\mathcal{M}}, pk_{\mathcal{M}}, pk_{\mathcal{B}}, Params)$, $\mathcal{B}(pk_{\mathcal{M}}, Params)$)**:** an interactive protocol between $\mathcal{M}$ and $\mathcal{B}$. During the deposit, $\mathcal{B}$ receives (S, Π) from $\mathcal{M}$, checks that it is fresh and that Π is correct. If not, $\mathcal{B}$ outputs $\perp_1$. Else $\mathcal{B}$ computes 2^ℓ serial numbers $\widetilde{S}_1, \ldots, \widetilde{S}_{2^\ell}$ from (S, Π) and $Params$. If one of the serial number $(\widetilde{S}_i, S', \Pi')$ already belongs to $\mathcal{L}$, then the bank outputs $(\perp_2, S, \Pi, S', \Pi')$. Otherwise, $\mathcal{B}$ adds $(\widetilde{S}_i, S, \Pi)$, $1 \leq i \leq 2^\ell$, to its list $\mathcal{L}$ of spent coins, credits $\mathcal{M}$'s account, and returns $\mathcal{L}$. $\mathcal{M}$'s output is OK or $\perp$.
- `Identify`($(S_1, \Pi_1), (S_2, \Pi_2), Params$)**:** a deterministic algorithm executed by $\mathcal{B}$ that outputs a public key $pk_{\mathcal{U}}$ and a proof Π_G. If $\mathcal{M}$s who had submitted Π_1 and Π_2 are not malicious, then Π_G is evidence that $pk_{\mathcal{U}}$ is the registered public key of a user that double-spent a coin.
- `VerifyGuilt`($pk_{\mathcal{U}}, \Pi_G, Params$)**:** a deterministic algorithm executed by any actor that outputs 1 if the proof is correct and 0 otherwise. This verification permits anyone to be sure that the user with public key $pk_{\mathcal{U}}$ is guilty of double-spending a coin.

2.2 Notions of Security

In the following, it is assumed that the overlying experiment has run the algorithm `ParamKeyGen` on input k to obtain the parameters $Params$.

- **Unforgeability.** Let $\mathcal{A}$ be a p.p.t. Turing Machine. At the start of the game, $\mathcal{A}$ is given the public key $pk_{\mathcal{B}}$ and $Params$. Suppose that $\mathcal{A}$ interacts K times with an honest bank during withdrawal protocols, then the probability that the number of valid coins that has been spent is at least $2^L K + 1$ is negligible.
- **Unlinkability.** Let $\mathcal{A}$ be a p.p.t. Turing Machine. At the start of the game, $\mathcal{A}$ is given the key pair $(pk_{\mathcal{B}}, sk_{\mathcal{B}})$ and $Params$. At the end, $\mathcal{A}$ chooses two honest users 0 and 1. A bit b is secretly and randomly chosen. Then, a spending protocol is played by $\mathcal{A}$ with user b (it is assumed that both honest users still have unspent coins). Finally, $\mathcal{A}$ outputs a bit b'. We require that for every $\mathcal{A}$ playing this game, the probability that $b = b'$ differs from $1/2$ by a fraction that is at most negligible.
- **Identification of double-spenders.** Let $\mathcal{A}$ be a p.p.t. Turing Machine. At the start of the game, $\mathcal{A}$ is given the public key $pk_{\mathcal{B}}$ and $Params$. The probability that a `Deposit` protocol between an honest merchant and an honest bank outputs $(\perp_2, S, \Pi, S', \Pi')$ such that the output of `Identify` algorithm on inputs (S, Π, S', Π') is not the public key $pk_{\mathcal{U}}$ of a corrupted user is negligible.
- **Exculpability.** Let $\mathcal{A}$ be a p.p.t. Turing Machine. At the start of the game, $\mathcal{A}$ is given the key pair $(pk_{\mathcal{B}}, sk_{\mathcal{B}})$ and $Params$. During the game, $\mathcal{A}$ interacts with honest users to

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

supply them coins. At the end, $\mathcal{A}$ constructs two spent coins (S_1, Π_1) and (S_2, Π_2). The probability that the outputs of the `Identify` algorithm on inputs (S_1, Π_1) and (S_2, Π_2) is the public key $pk_{\mathcal{U}}$ of an honest user together with a valid proof Π_G is negligible.

Remark 1. Notice that the exculpability property implies that the bank cannot create withdrawals for which the user has not participated. We don't need any extra security property, such as the proposal in [28].

3 General Description

In an anonymous e-cash system without a trusted third party, spending a single coin consists in generating a valid serial number S to allow double-spending detection and a valid security tag T masking the identity of the spender. The spender has to prove that S and T are well-formed without giving any information about his identity. In particular, the identity of the spender must be recovered only in case of double-spending by using the security tag T.

The main motivation of divisible e-cash is to provide a method to withdraw or spend several coins more efficiently than repeating several times a single withdrawal or spending protocol. We provide a general approach to construct divisible e-cash systems strongly unlinkable and truly anonymous (the user identity can be recovered only in case of fraud). This construction can be applied using several basic cryptographic tools.

3.1 Truly Anonymous E-cash Scheme based on Binary Trees

The general principle of our construction is derived from the classical binary tree approach [21, 9, 20] with slight modifications. Each divisible coin of monetary value 2^L is assigned to a binary tree of $L + 2$ levels. The tree root (level 0) with monetary value 2^L is assigned to a serial number denoted by $N_{0,0}$. Any other node has a monetary value corresponding to half of the amount of its parent node, except for the leaves that have no monetary value: they are "dead" leaves. For every level i, $0 \leq i \leq L$, the 2^i nodes are assigned serial numbers denoted by $N_{i,j}$ with $1 \leq j \leq 2^i$, except for the "dead" leaves that are not related to any serial number. Any *divisible* e-cash system should verify the divisibility rule.

Definition 1. *When a node N is used, none of descendant and ancestor nodes of N can be used, and no node can be used more than once.*

This rule is satisfied if, and only if, over-spending is protected. The general principle of our proposal consists in using a single master serial number from which several serial numbers can be derived. Thus, each node of the tree, which includes the leaves, is also related to a particular value called a *tag key*. During the spending protocol, the identity of the spender is encrypted with a tag key in such a way that the decryption key can be derived only in case of a double-spending. Using the binary tree approach, each node of the tree is related to a tag key with the following properties.

- The root tag key and the identity of the user are signed (in a blind manner) by the bank during the withdrawal protocol.

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

– From the tag key of a node N, it is possible for everyone to compute the tag keys related to the descendant nodes of N. It consequently exists a public deterministic function $\mathcal{F}$ that takes as input a tag key K_{i,b_0} (where i is the level of the targeted node in the tree and $b_0 \in \{0,1\}$ depends on the position of K in the tree[3]), a bit b (0 for left and 1 for right) and possibly some public parameters $Params$ and that outputs a new tag key $K_{i+1,b}$.

$$\mathcal{F} : (K_{i,b_0}, b, Params) \longrightarrow K_{i+1,b} = \mathcal{F}(K_{i,b_0}, b, Params).$$

– From the tag key of a node, it is impossible (without the knowledge of the root tag key) to compute a tag key which is not related to a descendant of the targeted node.
– The serial number of a particular node is the concatenation of the two children tag keys. Notation is given in Figure 1.

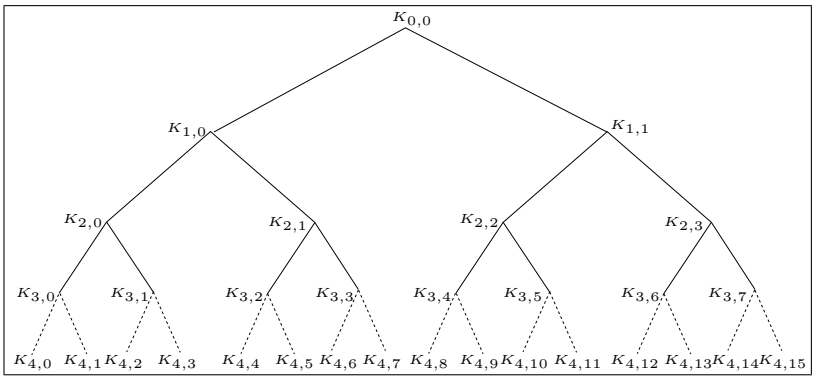

Fig. 1. General principle - Tree of keys

During the spending protocol, the user computes the tag key of the node he wants to spend. This tag key is used to compute the security tag, i.e. the encryption of the spender identity. This encryption should be verifiable and should include randomness. This randomness should be provided by the merchant to ensure the freshness of the spending, i.e., to prevent merchant from sending twice the same coin to the bank. The user also computes the tag keys related to the two direct descendants of the spent node. The concatenation of these two keys is the serial number of the spent coin. This serial number is transmitted during the spend protocol. Later, the bank will compute all the serial numbers of the leaves of the tree in order to detect a possible double-spending. If a double-spending is detected, then the bank has access to the encryption of the identity (from one spending) and the corresponding decryption key (from the other spending). Then, the bank can easily find the identity of the cheater.

Example 1. Assume $\mathcal{U}$ wants to spend four coins. Then, $\mathcal{U}$ selects four unitary coins, e.g. those associated to the node $K_{1,0}$. The user $\mathcal{U}$ sends to $\mathcal{M}$ the values $T = E_{K_{1,0}}(Id, R)$, $LK = K_{2,0}$, $RK = K_{2,1}$, and $S = LK \| RK$. The random value R used in the encryption scheme is computed using values sent by the merchant. The user must also prove that the coins are signed by the bank and that it will be possible to identify a double-spender. Consequently, the spending protocol consists also in computing a zero-knowledge proof of knowledge Φ that corresponds to the predicates:

[3] $b_0 = 0$ if and only if the targeted node belongs to the left subtree of its ancestor.

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

– T is well-formed, i.e. $E_{K_{1,0}}(Id, R)$ has been computed using:
  • the tag key $K_{1,0}$ derived using $\mathcal{F}$ on inputs the root tag key $K_{0,0}$ signed by the bank,
  • the random R that has been chosen by the merchant,
  • the identity Id signed by the bank.
– LK and RK are well-formed, i.e., $K_{2,0}$ and $K_{2,1}$ are both derived from $K_{1,0}$ using $\mathcal{F}$.
– If LK and RK are well-formed, this implies that the serial number S is also well-formed.

To construct a truly anonymous divisible e-cash system, it is then necessary to provide a function $\mathcal{F}$, a verifiable encryption scheme E and a proof Φ. We give an example in Section 4.

3.2 Useful Tools

Proofs of Knowledge. We use zero-knowledge proofs of knowledge constructed over a cyclic group $\mathcal{G}$ either of prime order q or of unknown order: proof of equality of two known representations [10, 6], proofs of knowledge of a discrete logarithm [26, 17], of a representation, of a double discrete logarithm $PK(\alpha/z = g^{\alpha} \wedge y = g_1^{g_2^{\alpha}})$ [27, 20], proof of the "or" statement $PK(\alpha/T_1 = h_1^{\alpha} \vee T_2 = h_2^{\alpha})$ [11, 25]. We also need a proof of knowledge of one out of two double discrete logarithm $PK(\alpha/T_1 = g^{h_1^{\alpha}} \vee y = g^{h_2^{\alpha}})$ which is a combination of the two above proofs. These proofs can also be used non interactively by using the Fiat-Shamir heuristic [16].

Camenisch-Lysyanskaya Signature Schemes. These signature schemes are proposed in [5] with in addition some specific protocols:

– an efficient protocol between a user $\mathcal{U}$ and a signer $\mathcal{S}$ that permits $\mathcal{U}$ to obtain from $\mathcal{S}$ a signature σ of some commitment C on values $(x_1, \ldots, x_l)$ unknown from $\mathcal{S}$. $\mathcal{S}$ computes $\texttt{CLSign}(C)$ and $\mathcal{U}$ gets $\sigma = \texttt{Sign}(x_1, \ldots, x_l)$ that can be verified by $\texttt{Verif}(\sigma, (x_1, \ldots, x_l)) = 1$.
– an efficient proof of knowledge of a signature on committed values, denoted by $PK(\alpha_1, \ldots, \alpha_l, \beta/\beta = \texttt{Sign}(\alpha_1, \ldots, \alpha_l))$.

These constructions are quite close to group signature schemes. This is the case of the two following examples, one based on the ACJT signature scheme [1], secure under the Flexible RSA assumption [15], and the other based on the BBS one [2], secure under the q-SDH assumption [2].

4 Divisible E-cash System $\mathcal{DCS}$

We apply the general construction presented in Section 3.1 to the binary tree used in the system described in [20]. The function $\mathcal{F}$ is chosen to be the modular exponentiation. For each level i, there are three linked generators $g_{i,0}$ for "left", $g_{i,1}$ for "right" and $g_{i,2}$ to compute the security tag. For a node at level $i-1$ represented by the tag key denoted by K_{i-1,b_0}, the tag key of, e.g. the left children, is $K_{i,0} = g_{i,0}^{K_{i-1,b_0}}$. For the tag key $K_{i,b}$ and a random value R computing using merchant data, the encryption of the user identity $pk_{\mathcal{U}}$ is defined to be $pk_{\mathcal{U}} g_{i+1,2}^{K_{i,b} \cdot R}$. In the following, we assume that $\mathcal{H}$ is a collision-resistant hash function.

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

4.1 Setup

We consider a group $\mathcal{G}$ of order $o_{\mathcal{G}}$. The elements h_0,h_1, h_2 are random generators of $\mathcal{G}$. $\mathcal{G}_1 = \langle g_1 \rangle$ is a subgroup of $\mathbb{Z}^*_{o_{\mathcal{G}}}$ and each group $\mathcal{G}_i = \langle g_i \rangle$ must be a subgroup of $\mathbb{Z}^*_{o_{i+1}}$ where o_{i+1} is the order of $\mathcal{G}_{i+1}$. For example [20], it is possible to take $\mathcal{G}_i$ as a subgroup of $\mathbb{Z}^*_{o_i+1}$ for the prime $o_{i+1} = 2o_i + 1$ with all i. As a consequence, the group $\mathcal{G}_i$ is related to the level i of the tree. The following generators are randomly chosen: g in $\mathcal{G}$, $g_{1,0}$, $g_{1,1}$, $g_{1,2}$ in $\mathcal{G}_1$, $g_{2,0}$, $g_{2,1}$, $g_{2,2}$ in $\mathcal{G}_2$, $\ldots$, $g_{L+1,0}$, $g_{L+1,1}$, $g_{L+1,2}$ in $\mathcal{G}_{L+1}$ whose discrete logarithms to the base $g_1, g_2, \ldots, g_{L+1}$ are unknown, respectively. All these data compose the public parameters $Params$ of the system and can be computed by the bank. The bank $\mathcal{B}$ computes the key pair $(sk_{\mathcal{B}}, pk_{\mathcal{B}})$ of a Camenisch-Lysyanskaya signature scheme that will permit it to sign a divisible coin, using the `CLSign` algorithm.

A user $\mathcal{U}$ (resp. a merchant $\mathcal{M}$) can compute its key pair $(sk_{\mathcal{U}}, pk_{\mathcal{U}})$ (resp. $(sk_{\mathcal{M}}, pk_{\mathcal{M}})$) by choosing randomly $u \in [0, o_{\mathcal{G}}[$ (resp. $m \in [0, o_{\mathcal{G}}[$) and computing g^u (resp. g^m). The value u (resp m) is the private key $sk_{\mathcal{U}}$ (resp. $sk_{\mathcal{M}}$) and g^u (resp. g^m) is equal to the public key $pk_{\mathcal{U}}$ (resp. $pk_{\mathcal{M}}$).

4.2 Withdrawal Protocol

During a withdrawal protocol, $\mathcal{U}$ interacts with $\mathcal{B}$. $\mathcal{U}$'s inputs are $pk_{\mathcal{B}}$, $sk_{\mathcal{U}}$, $pk_{\mathcal{U}}$ and $Params$, and $\mathcal{B}$'s inputs are $pk_{\mathcal{U}}$, $sk_{\mathcal{B}}$, $pk_{\mathcal{B}}$ and $Params$.

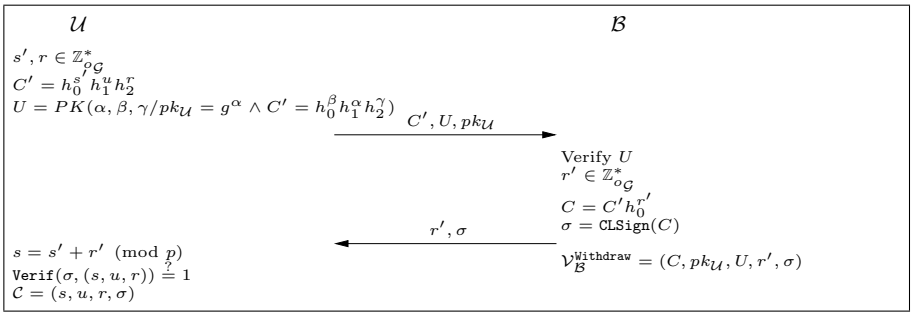

Fig. 2. Withdrawal protocol

The withdrawal protocol permits $\mathcal{U}$ to obtain a new divisible coin by interacting with $\mathcal{B}$ as described in Figure 2. A divisible coin corresponds to a (blind) CL signature done by $\mathcal{B}$ on a secret s and the secret key u of $\mathcal{U}$. Both $\mathcal{U}$ and $\mathcal{B}$ participate to the randomness of the secret s. At the end of the `Withdraw` protocol, $\mathcal{U}$ gets a divisible coin $\mathcal{C} = (s, u, r, \sigma = \mathtt{Sign}(s, u, r))$.

4.3 Spending Protocol

When $\mathcal{U}$ wants to spend to $\mathcal{M}$ a sub-coin of value 2^ℓ ($\ell = L - i$) from his divisible coin $\mathcal{C}$, he chooses an unspent node of the level i, e.g. the node $N_{i,j}$. A spending protocol of the node $N_{i,j}$ consists in the following.

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

1. $\mathcal{M}$ sends to $\mathcal{U}$ a random value $rand$ and $\mathcal{U}$ computes $R = \mathcal{H}(pk_{\mathcal{M}} \| rand)$.
2. $\mathcal{U}$ randomly chooses $\tilde{g}, \tilde{h} \in \mathcal{G}$, $\tilde{g}_1 \in \mathcal{G}_1$, $\tilde{g}_2 \in \mathcal{G}_2$, $\ldots$, $\tilde{g}_{i+1} \in \mathcal{G}_{i+1}$.
3. $\mathcal{U}$ executes the algorithm presented in Figure 3 (in pseudo-code) for the node $N_{i,j}$, outputting the values[4] $(\widetilde{V}_0, \ldots, \widetilde{V}_i, V)$, using the path from the root tree to the node $N_{i,j}$. Next,

Input: i, j
Output: $(\widetilde{V}_0, \ldots, \widetilde{V}_i, V)$

$\tilde{r} \leftarrow$ `Rand()`, $V \leftarrow g^s$, $\widetilde{V}_0 \leftarrow \widetilde{g}^s \tilde{h}^{\tilde{r}}$, `CurrentNode` $\leftarrow root$
If $i = 0$, **then** return $(\widetilde{V}_0, V)$
$a \leftarrow 1, b \leftarrow 2^i$
For $k = 1$ to i
  $\widetilde{V}_k \leftarrow \widetilde{g}_k^V$
  If $a \leq j \leq a + (b - a - 1)/2$, **then** \\ $N_{i,j}$belongs to `leftSubTree(CurrentNode)`
    $V \leftarrow (g_{k,0})^V$, $b \leftarrow a + (b - a - 1)/2$ \\ `CurrentNode` $\leftarrow$ `leftSon(CurrentNode)`
  Else \\ $N_{i,j}$ belongs to `rightSubTree(CurrentNode)`
    $V \leftarrow (g_{k,1})^V$, $a = a + (b - a + 1)/2$ \\ `CurrentNode` $\leftarrow$ `rightSon(CurrentNode)`
 return $(\widetilde{V}_0, \ldots, \widetilde{V}_i, V)$

Fig. 3. Spending protocol - Computation of V

$\mathcal{U}$ computes the security tag: $LK = g_{i+1,0}^V$, $RK = g_{i+1,1}^V$, $T = pk_{\mathcal{U}} g_{i+1,2}^{V \cdot R}$ and $S = LK \| RK$.

Example 2. Assume $\mathcal{U}$ wants to spend four coins (the same as in Example 1. The user $\mathcal{U}$ sends to the merchant $\mathcal{M}$ the values $LK = g_{2,0}^{g_{1,0}^{g^s}}$, $RK = g_{2,1}^{g_{1,0}^{g^s}}$, $T = pk_{\mathcal{U}}(g_{2,2}^{R \cdot g_{1,0}^{g^s}})$ and $S = LK \| RK$ since $V = g_{1,0}^{g^s}$.

4. $\mathcal{U}$ proves to $\mathcal{M}$ the validity of LK, RK, T (and thus the validity of S) using a non-interactive zero-knowledge proof of knowledge of a signature of $\mathcal{B}$ on the values (s, u, r) and that the value LK, RK, T are correctly computed. This proof of knowledge is constructed from a zero-knowledge proof of knowledge using the Fiat-Shamir heuristic. This proof is as follows:

$$
\begin{aligned}
\Phi = PK\Big(&\sigma, s, u, r, \tilde{r}, \alpha_1, \ldots, \alpha_{i+1}, \beta \ / \\
&\sigma = \mathtt{Sign}(s, u, r) \ \wedge \ \widetilde{V}_0 = \widetilde{g}^s \tilde{h}^{\tilde{r}} \ \wedge \ \widetilde{V}_1 = \tilde{g}_1^{g^s} \ \wedge \ \widetilde{V}_1 = \tilde{g}_1^{\alpha_1} \ \wedge \\
&(\widetilde{V}_2 = \tilde{g}_2^{g_{1,0}^{\alpha_1}} \ \vee \ \widetilde{V}_2 = \tilde{g}_2^{g_{1,1}^{\alpha_1}}) \ \wedge \ \widetilde{V}_2 = \tilde{g}_2^{\alpha_2} \ \wedge \ \ldots \ \wedge \\
&(\widetilde{V}_{i+1} = \tilde{g}_{i+1}^{g_{i,0}^{\alpha_i}} \ \vee \ \widetilde{V}_{i+1} = \tilde{g}_{i+1}^{g_{i,1}^{\alpha_i}}) \ \wedge \ \widetilde{V}_{i+1} = \tilde{g}_{i+1}^{\alpha_{i+1}} \ \wedge \\
&LK = g_{i+1,0}^{\alpha_{i+1}} \wedge RK = g_{i+1,1}^{\alpha_{i+1}} \wedge T = pk_{\mathcal{U}} g_{i+1,2}^{R \cdot \alpha_{i+1}} \Big)
\end{aligned}
$$

5. $\mathcal{U}$ sends the spent coins (S, Π) to $\mathcal{M}$, with $\Pi = \{2^\ell, T, \Phi, R, \widetilde{V}_0, \ldots, \widetilde{V}_i\}$.

4.4 Deposit Protocol

When $\mathcal{M}$ wants to deposit a coin (S, Π) to $\mathcal{B}$, $\mathcal{M}$ just sends the coin (S, Π) to $\mathcal{B}$. The proof Π should include the monetary value 2^ℓ of the divisible coin, the security tag T, the proof

[4] The values $\widetilde{V}_0, \ldots, \widetilde{V}_i$ are computed to prove that the value V is well computed. See proof Φ below and [20].

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

of knowledge Φ and the random data R provided by the merchant. $\mathcal{B}$ checks the validity of Φ and the consistency with S. If (S, Π) is not a valid coin, $\mathcal{B}$ rejects the deposit. Else, $\mathcal{B}$ computes, from S, 2^ℓ serial numbers $\widetilde{S}_{k_1}, \ldots \widetilde{S}_{k_{2^\ell}}$ corresponding to the $2^{\ell+1}$ dead leaves of the sub-tree. This is done by applying several modular exponentiation functions to S, using the right generators. $\mathcal{B}$ has to deal with 2^ℓ unitary coins $(\widetilde{S}_{k_j}, S, \Pi)$, $1 \leq j \leq 2^\ell$.

For every unitary coin $(\widetilde{S}_{k_j}, S, \Pi)$, $\mathcal{B}$ checks if there is already an entry $(\widetilde{S}_{k_j}, S', \Pi')$ in the database. If there is no entry in the database for the serial number $\widetilde{S}_{k_j}$, then $\mathcal{B}$ accepts the deposit of the coin $(\widetilde{S}_{k_j}, S, \Pi)$, credits the $pk_{\mathcal{M}}$'s account and add $(\widetilde{S}_{k_j}, S, \Pi)$ to the database of spent coins. Else, there is an entry $(\widetilde{S}_{k_j}, S', \Pi')$ in the database. Then, $\mathcal{B}$ checks the freshness of merchant randomness R in Π compared to Π'. If it not fresh, $\mathcal{M}$ is a cheat and $\mathcal{B}$ refused the deposit. If R is fresh, $\mathcal{B}$ accepts the deposit of the coin $(\widetilde{S}_{k_j}, S, \Pi)$, credits the $pk_{\mathcal{M}}$'s account and add $(\widetilde{S}_{k_j}, S, \Pi, S', \Pi',)$ to the list of double-spenders. For every entry of the database of double-spenders, $\mathcal{B}$ will executes the `Identify` algorithm.

4.5 Identify

Assume that a double detection has been done. Then $\mathcal{B}$ knows two accepted spending $(2^{I_1}, S_1 = LK_1 \| RK_1, T_1, R_1, \Phi_1)$ with $I_1 = L - i_1$ and $(2^{I_2}, S_2 = LK_2 \| RK_2, T_2, R_2, \Phi_2)$ with $I_2 = L - i_2$ such that e.g. S_1 is an ancestor of S_2 or $S_1 = S_2$. If $S_1 = S_2$ then the bank can directly get the public key $pk_{\mathcal{U}}$ by computing $\left(T_1^{R_2} / T_2^{R_1}\right)^{1/(R_2 - R_1)} = pk_{\mathcal{U}}$. If S_1 is an ancestor of S_2, then the bank computes the masking value $g_{I_2+1,2}^{V_2}$ (s.t. $T_2 = pk_{\mathcal{U}} g_{I_2+1,2}^{R_2 \cdot V_2}$) from the knowledge of LK_1 and RK_1 and the path[5] from $N_{i_1}^{j_1}$ up to $N_{i_2}^{j_2}$ as described in Figure 4. Then, $\mathcal{B}$ computes

Input: i_1, j_1, i_2, j_2
Output: V_2

```
CurrentNode ← N_{i_1}^{j_1}
If N_{i_2}^{j_2} belongs to leftSubTree(CurrentNode), then
  V_2 ← LK_1; CurrentNode ← leftSon(CurrentNode);
Else
  V_2 ← RK_1; CurrentNode ← rightSon(CurrentNode);
For k = i_1 + 2 to i_2 do
   If N_{i_2}^{j_2} belongs to leftSubTree(CurrentNode) , then
     V_2 ← (g_{k,0})^{V_2}; CurrentNode ← leftSon(CurrentNode)
   Else
     V_2 ← (g_{k,1})^{V_2}; CurrentNode ← rightSon(CurrentNode)
  k = k + 1
 return V_2
```

Fig. 4. Identify protocol - Computation of V_2

the public key $pk_{\mathcal{U}}$ as follows: $(T_2)^{\frac{1}{R_2}} / g_{I_2+1,2}^{V_2} = pk_{\mathcal{U}}$.

[5] The values $N_{i_1}^{j_1}$ and $N_{i_2}^{j_2}$ are not know by $\mathcal{B}$ but $\mathcal{B}$ knows the path from $N_{i_1}^{j_1}$ up to $N_{i_2}^{j_2}$ since it knows the path used to compute the colliding serial numbers.

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

4.6 Verify Guilt

The algorithm `VerifyGuilt` can be executed by any actor from the parameters of the system *Params* and a proof Π_G. One can parse the proof Π_G as $\big((2^{\ell_1}, S_1, R_1, T_1, \Pi_1), (2^{\ell_2}, S_2, R_2, T_2, \Pi_2)\big)$ and next run `Identify` on these values. If the algorithm `Identify` returns a public key $pk_{\mathcal{U}}$, then one can check if Π_1 is consistent with $(2^{\ell_1}, S_1, R_1, T_1)$ and if Π_2 is consistent with $(2^{\ell_2}, S_2, R_2, T_2)$. If both are consistent then accept, else reject.

5 Security Arguments

In this section, we provide the Theorem that stipulates that the $\mathcal{DCS}$ scheme is a secure divisible e-cash system.

Theorem 1. *In the random oracle model, the $\mathcal{DCS}$ scheme is secure:*

- *If the CL signature scheme is unforgeable, then $\mathcal{DCS}$ is unforgeable.*
- *Under the DDH assumption, $\mathcal{DCS}$ is unlinkable.*
- *If the CL signature scheme is unforgeable, then $\mathcal{DCS}$ permits the identification of double-spenders.*
- *Under the DL assumption (and the Flexible RSA assumption if $\mathcal{DCS}$ relies on the ACJT scheme), $\mathcal{DCS}$ has the exculpability property.*

Proof. We have to show that $\mathcal{DCS}$ verifies all security properties.

Unforgeability. We want to show that if an adversary $\mathcal{A}$ is able to break the unforgeability of our construction, then it is possible to break the unforgeability of the CL signature scheme under adaptive chosen message attack.

We can interact with $\mathcal{A}$ during the withdrawal protocol by playing the role of an honest bank with access to the signature oracle. After each successful spending executed by $\mathcal{A}$, we extract, using standard technique, the values (u, s, r, σ) satisfying the relation embedded into the valid proof of knowledge Π. Since there are more spent coins than $\mathcal{A}$ can legitimely own, and since there is no detection of double-spending (by assumption), then it is necessary that, among all extracted values $(u_j, s_j, r_j, \sigma_j)$, one signature σ on a message $m = (s, u, r)$ is unknown and does not come from the signature oracle. Thus, this one more signature is a signature (forgery) in the CL's scheme on the message $m = (u, s, r)$.

As the CL signature scheme is proven secure against adaptive chosen message attacks under the Flexible RSA assumption (if the scheme relies on the ACJT scheme) or the q-SDH (if the scheme relies on the BBS scheme), it follows that $\mathcal{A}$ cannot succeed with non negligible probability.
Because our proof requires rewinding to extract s' and r from an adversary $\mathcal{A}$, our proof is valid only against sequential attacks. Indeed, in a concurrent setting where the attacker is allowed to interact with the bank in an arbitrarily interleaving manner, our machine may be forced to rewind an exponential number of times. This drawback can be overcome by using for instance well-know techniques [12] which would require from the user to encrypt s' and r in a verifiable manner [7].

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

Unlinkability. We want to show that if an adversary $\mathcal{A}$ is able to break the unlinkability of our construction, then it is possible to break an instance of the Diffie-Hellman problem. In fact, we use a variant of the Diffie-Hellman problem, called Matching Multi Diffie-Hellman (MMDH) problem, and we prove in Appendix A that if someone is able to solve the MMDH problem, then it is possible to solve a given instance of the DDH problem.

We can interact with $\mathcal{A}$ during the withdraw protocol by playing the role of an honest user except for the two first interactions where we use the MMDH instance. During spending protocols, we can interact with $\mathcal{A}$ by playing the role on an honest user, except when the divisible coin corresponds to one of the two divisible coins associated with the MMDH instance to be solved.

We can win the game when $\mathcal{A}$ chooses the two first users (corresponding to the MMDH instance) and thus use the MMDH instance during the execution of the final spend. If $\mathcal{A}$ does not choose users i_0 and i_1 for the challenge we need to play again the game.

We denote by q_U the average number of users created by $\mathcal{A}$. Our success probability is $\epsilon' = 1 - (1 - (1/2 + \epsilon/2))^{q_U} \equiv 1/2 + q_U\epsilon/2$ within polynomial $\mathcal{T}' = q_U\mathcal{T} + \tau$, where τ is polynomial.

Remark 2. In the simulation, we use the instance of the MMDH problem to interact with $\mathcal{A}$. We also need to choose a value for the bit b. If our choice of b is correct, then there is no problem and we will be able to conclude with the advantage ϵ of $\mathcal{A}$. If this choice is uncorrect, $\mathcal{A}$ has a probability exactly equal to $1/2$ as ours. Repeating the game many times, our success probability of solving the MMDH instance is greater than $1/2$.

Identification of Double-spenders. We want to show that if an adversary $\mathcal{A}$ is able to break the identification of double-spenders property, then it is possible to break the unforgeability of the CL signature scheme.

We have access to a signature oracle taking as input a commitment and outputting a signature on committed values. We interact with $\mathcal{A}$ during withdrawal protocols by playing the role of an honest bank. We also interact with $\mathcal{A}$ during spending protocols playing the role of the merchant. Note that there is no honest users in the game. After each successful spending executed by $\mathcal{A}$, we extract the values (u, s, r, σ) satisfying the relation embedded into the valid proof of knowledge Π. When there is a double-spending, i.e. $(\perp_1, S_1, \Pi_1), (S_2, \Pi_2)$, that means that there exist a valid serial number $\tilde{S}$ which can be computed from both S_1 and S_2. Furthermore, the proof Π_1 is consistent with S_1 and the proof Π_2 is consistent with S_2 and $R_1 \neq R_2$ where R_1 is the random chosen by the merchant in Π_1 and R_2 is the random chosen by the merchant in Π_2. Both Π_1 and Π_2 contains a proof of knowledge of a signature of the bank on the master serial number seed s used to generate S_1, S_2 and $\tilde{S}$. Thus, these two signatures σ_1 and σ_2 are such that at least one of the two is different from the signatures obtained during the execution of the `Withdrawal` protocols submitted to the signature oracle. This signature (σ_1 or σ_2) is thus a forgery on CL signature scheme. As the CL signature scheme is proven secure against adaptive chosen message attacks, it follows that $\mathcal{A}$ cannot succeed with non-negligible probability.

Exculpability. The adversary $\mathcal{A}$ wins the game if he can falsely accuse an honest user of a double-spending. This means that the adversary can interact with honest users to obtain

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

spending from them and he wins if he can produce one spend (S', T', Π') related to a valid one (S', T', Π') and such that the output of $\texttt{Identify}((S, T, \Pi), (S', T', \Pi'))$ is a public key $pk_{\mathcal{U}}$ of a honest user (with non negligible probability).

The security proof of the exculpability involves forking lemma-like technique for an attacker that exploits both valid spending played by honest users and valid withdrawals played by honest users when the extractability of the RO proofs-of-knowledge relies on the DL assumption in order to falsely accuse an honest user. If the Camenisch-Lysysanskaya scheme of the withdrawal protocol uses a group of unknown order, then the exculpability relies on both the DL assumption for an attacker that exploits valid spendings played by honest users in order to falsely accuse an honest user, and on the factorization assumption to ensure the non-malleability and the soundness of the proof of knowledge Φ (see [3]).

6 Conclusion

In this paper, we present the first off-line divisible e-cash scheme that provides strong unlinkability and truly anonymity. We introduced the idea of using a security tag in a divisible e-cash scheme. The anonymity of users is achieved without impacting the performance of the spending protocol and without using a trusted third party. The spending protocol exploits the binary structure underlying the divisible coin in order to get an efficient spending protocol. However, even if the new scheme permits the spending of multiple coins at a time, it uses double-exponentiation proofs for the spending phase which is still a little expensive. Thus, for a small number of coins at a time, the spending is still expensive. Another possible improvement for the scheme could be to find a method to detect double spending without computing 2^L serial numbers for a divisible coin of monetary value 2^L.

Acknowledgements

We are grateful to Pascal Paillier and Jacques Traoré for their suggestions of improvement, and to Serge Fehr and anonymous referees for their valuable comments. We also wish to mention that a similar work has been independently done by Jan Camenisch, Markulf Kohlweiss, Anna Lysyanskaya and Maria Meyerovich.

References

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A Practical and Provably Secure Coalition-resistant Group Signature Scheme. Advances in Cryptology - Crypto'00, volume 1880 of LNCS, pages 255-270, 2000.
2. D. Boneh, X. Boyen and H. Shacham. Short Group Signatures using Strong Diffie Hellman. Advances in Cryptology - Crypto'04, volume 3152 of LNCS, pages 41-55, 2004.
3. F. Boudot and J. Traoré. Efficient Publicly Verifiable Secret Sharing Schemes with Fast or Delayed Recovery. ICISC'99, volume 1726 of LNCS, pages 87-102, 1999.
4. J. Camenisch, S. Hohenberger, and A. Lysyanskaya. Compact E-cash. Advances in Cryptology - Eurocrypt'05, volume 3494 of LNCS, pages 302-321, 2005.
5. J. Camenisch and A. Lysyanskaya. Signature Schemes and Anonymous Credentials from Bilinear Maps. Advances in Cryptology - Crypto'04, volume 3152 of LNCS, pages 56-72, 2004.
6. J. Camenisch and M. Michels. Proving in Zero-knowledge that a Number is the Product of Two Safe Primes. Advances in Cryptology - Eurocrypt'99, volume 1592 of LNCS, pages 107-122, 1999.

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

7. J. Camenisch and V. Shoup. Practical Verifiable Encryption and Decryption of Discrete Logarithms. In D. Boneh, editor, Advances in Cryptology - Crypto '03, volume 2729 of LNCS, pages 126-144. Springer, 2003.
8. S. Canard, A. Gouget, and E. Hufschmitt. A Handy Multi-coupon System. Applied Cryptography and Network Security - ACNS 2006, volume 3989 of LNCS, pages 66-81, 2006.
9. A.H. Chan, Y. Frankel, and Y. Tsiounis. Easy Come - Easy Go Divisible Cash. Advances in Cryptology - Eurocrypt'98, volume 1403 of LNCS, pages 561-575, 1998.
10. D. Chaum and T. Pedersen. Transferred Cash Grows in Size. Advances in Cryptology - Eurocrypt'92, volume 658 of LNCS, pages 390-407, 1993.
11. R. Cramer, I. Damgard, and B. Schoenmakers. Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols. Advances in Cryptology - Crypto'94, volume 839 of LNCS, pages 174-187, 1994.
12. I. Damgard. Efficient Concurrent Zero-knowledge in the Auxiliary String Model. Advances in Cryptology - Eurocrypt '00, volume 1807 of LNCS, pages 418-430, 2000.
13. S. D'Amingo, and G. Di Crescenzo. Methodology for Digital Money based on General Cryptographic Tools. Advances in Cryptology - Eurocrypt'94, volume 950 of LNCS, pages 156-170, 1994.
14. T. Eng, and T. Okamoto. Single-term Divisible Coins. Advances in Cryptology - Eurocrypt'94, volume 950 of LNCS, pages 306-319, 1994.
15. E. Fujisaki and T. Okamoto. Statistical Zero-knowledge Protocols to Prove Modular Polynomial Relations. Advances in Cryptology - Crypto'97, volume 1294 of LNCS, pages 16-30, 1997.
16. A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. Advances in Cryptology - Crypto'86, volume 263 of LNCS, pages 186-194, 1986.
17. M. Girault, G. Poupard and J. Stern. On the Fly Authentication and Signature Schemes Based on Groups of Unknown Order. Advances in Cryptology - Journal of Cryptology, Volume 19, Number 4. Pages 463-487, Springer-Verlag, 2006.
18. H. Handschuh, Y. Tsiounis, and M. Yung. Decision Oracles are Equivalent to Matching Oracles. Public Key Cryptography PKC '99, volume 1560 of LNCS, pages 276-289. Springer, 1999.
19. T. Nakanishi, M. Shiota, and Y. Sugiyama. An Unlinkable Divisible Electronic Cash with User's Less Computations using Active Trustees. ISITA 2002, 2002.
20. T. Nakanishi and Y. Sugiyama. Unlinkable Divisible Electronic Cash. ISW'00, pages 121-134, 2000.
21. T. Okamoto. An Efficient Divisible Electronic Cash Scheme. Advances in Cryptology - Crypto'95, volume 963 of LNCS, pages 438-451, 1995.
22. T. Okamoto, K. Ohta. Universal Electronic Cash. Advances in Cryptology - Crypto'91, volume 576 of LNCS, pages 324-337, 1992.
23. J.C. Pailles. New Protocols for Electronic Money. Advances in Cryptology - Asiacrypt'92, volume 718 of LNCS, pages 263-274, 1993.
24. D. Pointcheval and J. Stern. Security Arguments for Digital Signatures and Blind Signatures. Journal of Cryptology, Volume 13 - Number 3. Pages 361-396, Springer-Verlag, 2000.
25. A. De Santis, G. Di Crescenzo, G. Persiano, and M. Yung. On Monotone Formula Closure of SZK. FOCS 1994, pages 454-465, 1994.
26. C. P. Schnorr. Efficient Identification and Signatures for Smart Cards. Advances in Cryptology - Crypto'89, volume 435 of LNCS, pages 239-252, 1990.
27. M. Stadler. Publicly Verifiable Secret Sharing. Advances in Cryptology - Crypto'96, volume 1070 of LNCS, pages 190-199, 1996.
28. M. Trolin. A stronger definition for anonymous electronic cash. Cryptology ePrint Archive: Report 2006/241. 2006.

A Matching Multi Diffie-Hellman problem

The problem underlying the property of unlinkability for $\mathcal{DCS}$ is the *Matching Multi Diffie-Hellman problem (MMDH)*. We show that MMDH can be used to solve the *Decisional Diffie-Hellman problem (DDH)*.

Decisional Diffie-Hellman (DDH) problem: given a random generator $g \in \mathcal{G}$ where $\mathcal{G}$ has prime order and the values h^x, h^y, h^z, the problem consists in deciding if $xy = z$ or not.

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

Matching Multi Diffie-Hellman (MMDH) problem: let $\mathcal{H}$, $\mathcal{H}_1$ and $\mathcal{H}_2$ be groups of prime order such that $\mathcal{H}_1$ is a subgroup of $\mathbb{Z}_o^*$ where o is the order of $\mathcal{H}_2$. Given three random generators $h \in \mathcal{H}$, $h_1 \in \mathcal{H}_1$ and $h_2 \in \mathcal{H}_2$ and the values $h^{\alpha_0}, h^{\alpha_1}, h_2^{h_1^{\alpha_b}}$ and $h_2^{h_1^{\alpha_{\bar{b}}}}$ where $b \in \{0,1\}$, the problem consists in deciding if $b = 0$ or 1.

Decisional Multi Diffie-Hellman (DMDH) problem: let $\mathcal{H}$, $\mathcal{H}_1$ and $\mathcal{H}_2$ be groups of prime order such that $\mathcal{H}_1$ is a subgroup of $\mathbb{Z}_o^*$ where o is the order of $\mathcal{H}_2$. Given three random generators $h \in \mathcal{H}$, $h_1 \in \mathcal{H}_1$ and $h_2 \in \mathcal{H}_2$ and the values $h^{\alpha}, h_2^{h_1^{\beta}}$, the problem consists in deciding if $\alpha = \beta$ or not.

Derived Decisional Diffie-Hellman (DDDH) problem: given random generators $g_1, g_2 \in \mathcal{G}$ where $\mathcal{G}$ has prime order and the values g_1^a, g_2^b, the problem consists in deciding if $a = b$ or not.

The problem MMDH is at least as difficult as DMDH. In fact, the MMDH is the matching problem related to the decisional one DMDH. Therefore, Handschuh, Tsiounis and Yung show [18] that decision oracles are equivalent to matching oracles, which can be applied to our context.

The problem DMDH is at least as difficult as DDDH. Indeed, given an instance (g_1, g_2, g_1^a, g_2^b) of the DDDH problem, we can transform it into an instance $(h = g_1, h_1, h_2 = g_2, h^{\alpha} = g_1^a, h_1^{h_2^{\beta}} = h_1^{g_2^b})$ where h_1 is taken at random, of the DMDH problem. Thus, $a = b$ if and only if $\alpha = \beta$.

The problem DDDH is at least as difficult as DDH. Indeed, given an instance (g, g^x, g^y, g^z) of the DDH problem, we can transform is into an instance $(g_1 = g, g_2 = g^x, g_1 = g^x, g_2 = g^z)$ of the DDDH problem. Thus, we have $z = xy$ if and only if $a = b$.

We deduce that MMDH is at least as difficult as DDH.

Appeared in M. Naor (Ed.): EUROCRYPT 2007, LNCS 4515,pp. 482–497, 2007.

Anonymity in Transferable E-cash*

Sébastien Canard[1] and Aline Gouget[2]

[1] Orange Labs R&D, 42 rue des Coutures, BP6243, F-14066 Caen Cedex, France.
[2] Gemalto, 6, rue de la Verrerie, F-92190 Meudon, France.

Abstract. Regular cash systems provide both the *anonymity* of users and the *transferability* of coins. In this paper, we study the anonymity properties of transferable e-cash. We define two natural additional levels of anonymity directly related to transferability and not reached by existing schemes that we call *full anonymity (FA)* and *perfect anonymity (PA)*. We show that the FA property can be reached by providing a generic construction and that the PA's cannot. Next, we define two restricted perfect anonymity properties and we prove that it is possible to design a transferable e-cash scheme where a bounded adversary not playing the bank cannot recognize a coin he has already owned.

1 Introduction

Electronic cash systems aim at emulating regular cash. Users withdraw coins from a bank, and next pay merchants using them. Then, merchants deposit coins to the bank. Even if the property of *transferability* (i.e. received cash can be spend later without involving the bank) is seen as a fundamental property of regular cash, it is usually disregarded in the electronic setting. This lack of interest for transferable e-cash may be explained by the result given in [6] showing that it is impossible to transfer a coin without increasing its size. However, this apparent drawback is not always unacceptable for applications depending on the available amount of storage data. The main advantage of the transferability of e-cash would be the decrease of communications between the bank and all users.

The anonymity in (non transferable) e-cash systems is well-studied in the literature. When introducing the transferability property into e-cash schemes, new notions of anonymity appear that have not already been described in the literature. These new anonymity notions related to transferable e-cash are studied in this paper.

As far as we know, the first transferable e-cash schemes that provides a weak level of anonymity has been proposed in [10, 11]. The anonymity level is said *weak* since the spenders identities are protected but it is possible to link several spends of the same user.

Another method for transferring e-cash has been presented in [14, 6]. The anonymity level of this scheme is said *strong* since the spender identities are protected and it is not possible to link several spends of the same user. Very recently, two transferable e-cash schemes have been proposed in [5]. Both schemes improve the efficiency of [14, 6] by reducing the number of communications between the bank and users. One scheme offers a computational *strong* anonymity while the other one offers an unconditional *strong* anonymity. However, none of these schemes offers a "perfect" anonymity of spends since it is always possible for an adversary to recognize a coin that he has previously seen being spent.

* This work has been financially supported by the European Commission through the IST Program under Contract IST-2002-507932 ECRYPT and by the French Agence Nationale de la Recherche and the TES Cluster under the PACE project.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

There is a gap between the highest level of anonymity achieved by the transferable e-cash schemes of the state-of-the-art and the impossibility result given in [6] showing that transferable e-cash cannot fulfils an *unconditional* "perfect" anonymity since an unbounded payer can always recognize his own money if he sees it later in a payment.

In this paper, we contribute to reduce this gap, in one hand by showing the possibility for an e-cash system to fulfil higher levels of anonymity, and on the other hand by proving that a computational payer can always recognize his own money if he sees it later in a payment, meaning that transferable e-cash cannot provide a *computational* "perfect" anonymity.

In Section 2, we give formal definitions for transferable e-cash. In Section 3, we focus on the security properties related to anonymity and we introduce two new properties: the *Full Anonymity (FA)* meaning that the adversary $\mathcal{A}$ is not able to recognize a coin he has already observed during a spending between honest users, and the *Perfect Anonymity (PA)* meaning that $\mathcal{A}$ is not able to decide whether or not he has already owned a coin he is receiving. In Section 4, we show that a transferable e-cash scheme can fulfil the FA property by providing a generic construction, and that no scheme can fulfil the PA property by improving the impossibility result given in [6]. In Section 5, we give evidence that it is possible to design a PA scheme if we assume that the (not unbounded) adversary is not the bank. Finally, we conclude in Section 6.

2 Transferable E-cash

In this section, we define the algorithms of transferable e-cash, the variables and oracles used by the adversaries, and the classical security properties that are not related to anonymity; anonymity properties are defined in Section 3. Note that our model can easily be extended to wallets by using the compact e-cash techniques [2].

2.1 Algorithms

A transferable e-cash system involves two types of player: a bank $\mathcal{B}$ and a user $\mathcal{U}$. A coin is represented by an identifier Id and some values π needed to prove its validity.

– $\texttt{ParamGen}(k)$ is a probabilistic algorithm that outputs the parameters of the system Par (including the security parameter k).

– $\texttt{BKeyGen}(Par)$ (resp. $\texttt{UKeyGen}(Par)$) is a probabilistic algorithm executed by $\mathcal{B}$ (resp. $\mathcal{U}$) that outputs the key pair $(sk_{\mathcal{B}}, pk_{\mathcal{B}})$ (resp. $(sk_{\mathcal{U}}, pk_{\mathcal{U}})$).

– $\texttt{Withdraw}(\mathcal{B}(sk_{\mathcal{B}}, pk_{\mathcal{B}}, pk_{\mathcal{U}}, Par),\ \mathcal{U}(sk_{\mathcal{U}}, pk_{\mathcal{U}}, pk_{\mathcal{B}}, Par))$ is an interactive protocol where $\mathcal{U}$ withdraws from $\mathcal{B}$ one coin. At the end, $\mathcal{U}$ either gets a coin $C = (Id, \pi)$ and outputs OK, or outputs $\perp$. The output of $\mathcal{B}$ is either its view $\mathcal{V}_{\mathcal{B}}^{\mathtt{W}}$ of the protocol (including $pk_{\mathcal{U}}$), or $\perp$.

– $\texttt{Spend}$ $(\mathcal{U}_1(Id, \pi, pk_{\mathcal{U}_2}, Par), \mathcal{U}_2(sk_{\mathcal{U}_2}, pk_{\mathcal{B}}, Par))$ is an interactive protocol where $\mathcal{U}_1$ gives a coin to $\mathcal{U}_2$. At the end, either $\mathcal{U}_2$ outputs a coin $C = (Id_C, \pi_C)$ or $\perp$, and either $\mathcal{U}_1$ saves that C is a spent coin and outputs OK, or $\mathcal{U}_1$ outputs $\perp$.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

– `Deposit` $(\mathcal{U}(Id, \pi, sk_{\mathcal{U}}, pk_{\mathcal{U}}, pk_{\mathcal{B}}, Par), \mathcal{B}(sk_{\mathcal{B}}, pk_{\mathcal{B}}, pk_{\mathcal{U}}, \mathcal{L}, Par))$ is an interactive protocol where $\mathcal{U}$ deposits a coin (Id, π) at the bank $\mathcal{B}$. If (Id, π) is not consistent/fresh, then $\mathcal{B}$ outputs $\perp_1$. Else, if Id already belongs to the list of spent coins $\mathcal{L}$, then there is an entry (Id, π') and $\mathcal{B}$ outputs $(\perp_2, Id, \pi, \pi')$. Else, $\mathcal{B}$ adds (Id, π) to its list $\mathcal{L}$, credits $\mathcal{U}$'s account, and returns $\mathcal{L}$. $\mathcal{U}$'s output is OK or $\perp$.

– `Identify` (Id, π, π', Par) is a deterministic algorithm executed by $\mathcal{B}$ that outputs a public key $pk_{\mathcal{U}}$ and a proof Π_G. If the users who had submitted π and π' are not malicious, then Π_G is evidence that $pk_{\mathcal{U}}$ is the registered public key of a user that double-spent a coin.

– `VerifyGuilt`$(pk_{\mathcal{U}}, \Pi_G, Par)$ is a deterministic algorithm that can be executed by any actor. It outputs 1 if Π_G is correct and 0 otherwise.

2.2 Global Variables

The set of user's public (resp. secret) keys is denoted by $\mathcal{PK} = \{(i, pk_i) : i \in \mathbb{N}\}$ (resp. $\mathcal{SK} = \{(i, sk_i) : i \in \mathbb{N}\}$; $sk_i = \perp$ if user i is corrupted). The set of views of supplied coin by oracles is denoted by $\mathcal{SC}$ and the set of all coins owned by the oracles is denoted by $\mathcal{OC}$. The set of deposited electronic cash (corresponding to $\mathcal{L}$) is denoted by $\mathcal{DC}$.

2.3 Oracles

By convention, the name of an oracle corresponds to the action done by this oracle.

Creation and corruption of users. `Create`(i) executes `UKeyGen`$(Par) = (sk_i, pk_i)$, it defines $\mathcal{PK}[i] = pk_i$ and $\mathcal{SK}[i] = sk_i$ and it outputs pk_i. The oracle `Corrupt`(i, pk_i) defines $\mathcal{PK}[i] = pk_i$ and $\mathcal{SK}[i] = \perp$ and it outputs OK. `Corrupt`(i) outputs the value $\mathcal{SK}[i] = sk_i$ and it updates $\mathcal{SK}[i] = \perp$.

Withdrawal protocol. `Suppl`() plays the bank side and it updates $\mathcal{SC}$ by adding $\mathcal{V}_{\mathcal{B}}^{\mathtt{W}}$ with bit 1 to flag it as a corrupted coin. The oracle `Withd`(U) plays the user $\mathcal{U}$ side and it updates $\mathcal{OC}$ by adding the value (U, Id, π). The oracle `Withd&Suppl`(U) plays both sides and it updates $\mathcal{OC}$ as for `Withd`(U) and $\mathcal{SC}$ by adding $\mathcal{V}_{\mathcal{B}}^{\mathtt{W}}$ with flag 0. It outputs the communications between $\mathcal{B}$ and $\mathcal{U}$.

Spending protocol. The oracle `Rcv`(U_2) plays the role of user $\mathcal{U}_2$ with secret keys of user U_2 and it updates the set $\mathcal{OC}$ by adding a new entry (U_2, Id, π). The oracle `Spd`(U_1) plays the role of user $\mathcal{U}_1$ by spending a coin in $\mathcal{OC}$ owned by user U_1. It uses and updates the entry (U_1, Id, π) of $\mathcal{OC}$ as the `Spend` protocol describes it.
The oracle `Spd&Rcv`(U_1, U_2) plays the role of both $\mathcal{U}_1$ and $\mathcal{U}_2$ and it executes the spending of a coin owned by user U_1 to user U_2. It updates $\mathcal{OC}$ by adding the value (U_2, Id, π) and by flagging the coin as spent by U_1. It outputs all the communications of the spending.

Deposit protocol. `CreditAccount`() plays the role of the bank and it updates the set $\mathcal{DC}$. If the executed `Deposit` protocol outputs $(\perp_2, Id, \pi, \pi')$, then the oracle `CreditAccount` runs the algorithm `Identify` and outputs the result of this algorithm on inputs (Id, π, π'). The oracle `Depo`(U) plays the role of the user U during a `Deposit` protocol.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

2.4 Classical Security Properties not related to anonymity

Unforgeability. No collection of users can ever spend more coins than they withdrew.

Game. Let an adversary $\mathcal{A}$ be a p.p.t. Turing Machine that has access to the set of all user's public keys $\mathcal{PK}$, the bank's public key $pk_{\mathcal{B}}$ and Par. $\mathcal{A}$ can play as many times as he wants with the oracles: `Create`, `Corrupt`, `Suppl`, `Withd&Suppl`, `Spd`, `Spd&Rcv`, `Rcv` and `CreditAccount`.

Let q_W denote the number of successful queries to the oracle `Suppl`. Let q_R denote the number of successful queries to the oracle `Spd`. Let q_S be the number of successful queries to the oracle `Rcv`. The adversary $\mathcal{A}$ wins the game if, at any time during the game, we have $q_W + q_R < q_S$.

Identification of double-spenders. No collection of users can double-spend a coin twice without revealing one of their identities.

Game. Let an adversary $\mathcal{A}$ be a p.p.t. Turing Machine that has access to the $\mathcal{PK}$ global variable, the bank's public key $pk_{\mathcal{B}}$ and Par. $\mathcal{A}$ can play as many times as he wants with the oracles: `Create`, `Corrupt`, `Suppl`, `Withd&Suppl`, `Spd`, `Spd&Rcv`, `Rcv` and `CreditAccount`.

$\mathcal{A}$ wins the game if, at any time of the game, the oracle `CreditAccount` outputs ($\perp_2$, Id, π, π') and the output of the oracle `Identify` on inputs (Id, π, π') is not a public key related to a secret key $\perp$ in $\mathcal{SK}$.

Exculpability. The bank, even when cooperating with any collection of malicious users cannot falsely accuse users from having double-spent a coin.

Game. Let an adversary $\mathcal{A}$ be a p.p.t. Turing Machine that has access to the $\mathcal{PK}$ global variable, the bank's key pair $(pk_{\mathcal{B}}, sk_{\mathcal{B}})$ and Par. $\mathcal{A}$ can play as many times as he wants with the oracles: `Create`, `Corrupt`, `Withd`, `Spd`, `Spd&Rcv`, `Rcv` and `Depo`. At any time of the game, $\mathcal{A}$ outputs two spends (Id_1, π_1) and (Id_2, π_2). $\mathcal{A}$ wins the game if the outputs of the algorithm `Identify` on inputs (Id_1, π_1, π_2) is a public key pk such that the related secret key in $\mathcal{SK}$ is not $\perp$ together with a valid proof Π_G, and the output of the algorithm `VerifyGuilt` on inputs (pk, Π_G) is 1.

3 Anonymity Properties in Transferable E-cash

In this section, we focus on security properties related to anonymity, remembering usual ones and introducing our two new ones.

3.1 Overview

The *Weak Anonymity (WA)* and the *Strong Anonymity (SA)* properties are classical properties. Informally speaking,

- an e-cash scheme fulfils the WA property if an adversary cannot link a spending to a withdrawal. However, the adversary may know if two spends are done by the same user or not.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

– An e-cash scheme fulfils the SA property if it fulfils the WA property and if an adversary is not able to decide if two spends are done by the same user or not. However, the adversary may recognize a coin that he has already observed during previous spends.

We introduce two new anonymity properties directly related to the transferability property in e-cash that we call *Full Anonymity (FA)* and *Perfect Anonymity (PA)*. Informally speaking,

– an e-cash scheme fulfils the FA property if it fulfils the SA property and if an adversary is not able to recognize a coin that he has already observed during a spending between two honest users. However, the adversary may be able to recognize a coin he has already owned.
– An e-cash scheme fulfils the PA property if it fulfils the FA property and if an adversary is not able to decide whether or not he has already owned a coin he is receiving.

3.2 Description of the Game

Before defining the game, we need to recall that a transferred cash necessary grows in size [6]. This property may be exploited by the adversary $\mathcal{A}$ to win the game and break the anonymity property. Indeed, $\mathcal{A}$ may choose two users that do not own coins of the same size so as to distinguish which one is used at the end of the game.

Consequently, in the following game, we impose that the two challenged users i_0 and i_1 own coins of the same size and the coin used during the final call to the `Spd` oracle should be one of these coins.

Game. Let an adversary $\mathcal{A}$ be a p.p.t. Turing Machine that has access to the set of all user's public keys $\mathcal{PK}$.

1. $\mathcal{A}$ is given $(sk_{\mathcal{B}}, pk_{\mathcal{B}})$, Par and $\mathcal{A}$ can play with the oracles: `Create`, `Corrupt`, `Withd`, `Spd`, `Rcv`, `Spd&Rcv` and `Depo`.
2. At any time, $\mathcal{A}$ chooses two public keys $pk_{i_0}, pk_{i_1} \in \mathcal{PK}$, such that:
   (a) $\mathcal{SK}[i_0] \neq \perp$ and $\mathcal{SK}[i_1] \neq \perp$;
   (b) users i_0 and i_1 own coins of the same size;
   (c) users i_0 and i_1 have been used only by a set of *authorized oracles* that depends on the power of the adversary $\mathcal{A}$ (see below).
3. A bit b is secretly and randomly chosen and $\mathcal{A}$ plays with $\texttt{Spd}(i_b)$.
4. $\mathcal{A}$ outputs a bit b'.

3.3 Security Properties Related to Anonymity

We define four adversaries related to the four anonymity properties (WA, SA, FA and PA) that can be used to play the game described at Section 3.2. Indeed, the adversary is allowed to observe the transactions involving the coin that will be spend at step 3 with some specific restrictions depending on the anonymity property.

Definition 1 (Adversaries). *We denote by i_0 and i_1 the two users chosen by the adversary at Step 2 of the game described in Section 3.2.*

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

– *The adversary* $\mathcal{A}_{\mathtt{WA}}$ *is allowed to manipulate* i_0 *and* i_1 *only with the oracles:* `Create`, `Withd` *and* `Depo`.
– *The adversary* $\mathcal{A}_{\mathtt{SA}}$ *is allowed to manipulate* i_0 *and* i_1 *only with the oracles:* `Create`, `Withd`, `Spd`, `Spd&Rcv` *with the additional restriction that* i_0 *and* i_1 *do not play the role of* $\mathcal{U}_2$ *and* `Depo`.
– *The adversary* $\mathcal{A}_{\mathtt{FA}}$ *is allowed to manipulate* i_0 *and* i_1 *only with the oracles:* `Create`, `Withd`, `Spd`, `Spd&Rcv` *and* `Depo`.
– *The adversary* $\mathcal{A}_{\mathtt{PA}}$ *is allowed to manipulate* i_0 *and* i_1 *with all the oracles except the* `Corrupt` *oracle.*

Definition 2 (Anonymity properties). *A transferable e-cash system fulfils the property* $\mathcal{P} \in \{\mathtt{WA}, \mathtt{SA}, \mathtt{FA}, \mathtt{PA}\}$ *if for an adversary* $\mathcal{A}_{\mathcal{P}}$ *playing the game described at Section 3.2, the probability that* $b = b'$ *differs from* $1/2$ *by a fraction that is at most negligible.*

Remark 1. By construction, the anonymity properties are (exclusively) related one to the other as follows: $PA \Rightarrow FA \Rightarrow SA \Rightarrow WA$.

4 Study of Anonymity Properties

The schemes proposed in [14, 6, 5] fulfil both the WA and the SA properties but, as far as we know, the FA property (and consequently the PA property) is not achieved by any state of the art scheme. In this section, we first show that the FA property can be reached by providing a generic construction. Next, we prove that the PA property cannot be achieved.

4.1 Achieving the Full Anonymity Property

The difference between the SA game and the FA game is that the coin received at the challenge step by the adversary may have already been observed by $\mathcal{A}_{\mathtt{FA}}$ during a call to the `Spd&Rcv` oracle, whereas this case cannot happen during the SA game. The generic construction we propose is built upon an SA scheme. The key idea of our construction is to modify an SA scheme to get an FA scheme by protecting the communications of the spending protocol using the establishment of a unilateral authenticated secure channel between the receiver and the spender in order to prevent an active adversary to recognize a coin that he has previously seen being spent.

We assume that $\mathcal{S}$ is a transferable e-cash scheme that fulfils the SA property. From $\mathcal{S}$, we construct a transferable e-cash scheme $\mathcal{S}'$ that fulfils the FA property. We re-define only the spending protocol of $\mathcal{S}$ and we additionally use two building blocks: a secure symmetric encryption scheme $\mathcal{E} = (\mathtt{Enc}, \mathtt{Dec})$, and a unilateral authenticated key agreement protocol `KE` between two users (including a key-confirmation step) secure against an active adversary. In particular, `KE` is resistant to man-in-the-middle attacks.

A user $\mathcal{U}_1$ spends a coin to user $\mathcal{U}_2$ by first playing the `KE` protocol. At the end of the `KE` protocol, $\mathcal{U}_1$ and $\mathcal{U}_2$ share a unilateral authenticated session key K. Next, $\mathcal{U}_1$ and $\mathcal{U}_2$ play the `Spend` protocol of $\mathcal{S}$ by encrypting all the communications using the algorithms `Enc` and `Dec` with the common session key K.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

Theorem 1. *Under the assumptions that $\mathcal{S}$ fulfils the SA property, and* EK *and $\mathcal{E}$ are secure, the system $\mathcal{S}'$ fulfils the FA property.*

Sketch of Proof. Assume that $\mathcal{A}_{\mathtt{FA}}$ breaks the FA property by determining between users i_0 and i_1, which one is i_b. By definition $\mathcal{A}_{\mathtt{FA}}$ is not allowed to manipulate i_0 and i_1 with the oracle `Rcv` and thus $\mathcal{A}_{\mathtt{FA}}$ owns the coin of the challenge only at step 3 of the game. $\mathcal{A}_{\mathtt{FA}}$ should have seen it during the withdrawal of this coin (using the oracle `With`(U)) and possibly during spends between honest users (using the oracle `Spd&Rcv`).

By assumption ($\mathcal{S}$ fulfils the SA property), $\mathcal{A}_{\mathtt{FA}}$ cannot get the serial number of a coin involved in a withdrawal protocol. By construction of $\mathcal{S}'$, all communications related to the spending of a coin are encrypted with a unilateral authenticated ephemeral session key, which includes the communications related to a call to the oracle `Spd&Rcv`. Thus, $\mathcal{A}_{\mathtt{FA}}$ has no information about the identifier of the coin embedded into the spending ($\mathcal{A}_{\mathtt{FA}}$ may know the entry number of the coin in $\mathcal{OC}$ but not the serial number), except if $\mathcal{A}_{\mathtt{FA}}$ has succeeded in breaking either the security of `KE` to obtain the session key K or the security of $\mathcal{E}$ to decrypt the communications without knowing the decryption key. □

4.2 Impossibility of the Perfect Anonymity Property

It is known that a payer with unlimited computing power can always recognize his own money if he sees it later being spent [6], and thus the PA property cannot be achieved by an unlimited powerful adversary. In this section, we show that a bounded adversary $\mathcal{A}_{\mathtt{PA}}$, acting as the bank, can also win the anonymity game, meaning that the PA property cannot be achieved by transferable e-cash.

Attack against the PA Property. During the PA game, $\mathcal{A}_{\mathtt{PA}}$ creates users and corrupts some of them. At any time, $\mathcal{A}_{\mathtt{PA}}$ owns a set of valid coins $\{C_0, \cdots, C_l\}$ that he got from his interactions with the oracle `Spd`.

$\mathcal{A}_{\mathtt{PA}}$ chooses two honest users i_0 and i_1 such that they have no coins. Next $\mathcal{A}_{\mathtt{PA}}$ chooses two coins C_0 and C_1, spends coin C_0 to user i_0 and coin C_1 to user i_1 using the oracle `Rcv`. Then, $\mathcal{A}_{\mathtt{PA}}$ outputs i_0 and i_1, according to the PA game. The challenger next chooses at random a bit b and $\mathcal{A}_{\mathtt{PA}}$ plays a `Spend` protocol with the oracle `Spd` on input the user i_b. Acting as the bank, $\mathcal{A}_{\mathtt{PA}}$ then simply executes the `Deposit` algorithm for the coins C_b and C_0[3]. If a double spending is detected, then $\mathcal{A}_{\mathtt{PA}}$ outputs $b' = 0$ and he outputs $b' = 1$ otherwise. Thus, $\mathcal{A}_{\mathtt{PA}}$ can always succeed in guessing b.

5 Variants of the Perfect Anonymity Property

The attack described in Section 4.2 shows that the PA property cannot be achieved. In this section, we describe the two most natural ways to modify the PA game in order to prevent this attack. We define two distinct properties called $\mathrm{PA}_1{}^*$ and $\mathrm{PA}_2{}^*$ and show that both properties can be achieved. Next, we prove that there is no inclusion relation between the FA property and $\mathrm{PA}_1{}^*$ (resp. $\mathrm{PA}_2{}^*$).

[3] Even if this is a fraud, $\mathcal{A}_{\mathtt{PA}}$ can deposit the coin C_0 he has already spent.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

5.1 Additional Anonymity properties

The first possibility to make impossible the attack described in Section 4.2 is to prevent the adversary from receiving the coin C_b at Step 3 of the game described in Section 3.2. Then the coin C_b may have been manipulated by the adversary before Step 3 but the adversary only observes the spending of coin C_b between two honest users at Step 3.

Definition 3 (Adversary PA$_1$*). *The adversary $\mathcal{A}_{\mathtt{PA}_1^*}$ can manipulate the challenged users with all the oracles except the* `Corrupt` *oracle.*

Game of the PA$_1$* property. We modify the game described in Section 3.2 as follows. The steps 1 and 2 are unchanged. In step 3, the call to $\mathtt{Spd}(i_b)$ is replaced by a call to $\mathtt{Spd\&Rcv}(i_b, i)$ where i is a randomly chosen honest user.

The second possibility to avoid the attack against the PA property is to prevent the attacker to execute the deposit. We thus separate the power of the bank into two entities with distinct keys: $\mathcal{B}_W$ (resp. $\mathcal{B}_D$) is responsible of the withdraw (resp. the deposit) part. In the new property, the adversary is not allowed to control $\mathcal{B}_D$. Moreover, the `Deposit` and `Identify` protocols should be protected by a secret key of the bank $\mathcal{B}_D$. We should prevent the adversary from being able to simulate the deposit phase, as a user can do when this phase is based on public algorithms.

Definition 4 (Adversary PA$_2$*). *The adversary $\mathcal{A}_{\mathtt{PA}_2^*}$ can manipulate the challenged users with all the oracles except the oracles* `CreditAccount` *and* `Corrupt`*.*

Game of the PA$_2$* property. We modify the game described in Section 3.2 as follows. Only Step 1 is modified: `Withd` is replaced by `Suppl`, and `Depo` is replaced by `CreditAccount`.

Remark 2. Note that the discussion on public [13] or secret [2] `Deposit` and `Identify` is controversial in non-transferable e-cash definitions.

5.2 Studying the PA$_1$* Property

We first show that the PA$_1$* property can be achieved by proving that the scheme $\mathcal{S}'$ described in Section 4.1 is PA$_1$*.

Theorem 2. *The scheme $\mathcal{S}'$ described in Section 4.1 is PA$_1$*.*

Sketch of Proof. Assume that $\mathcal{A}_{\mathtt{PA}_1^*}$ breaks the PA$_1$* property of $\mathcal{S}'$ by determining between users i_0 and i_1 which one is i_b. Before Step 3 of the game, $\mathcal{A}_{\mathtt{PA}_1^*}$ has observed the withdrawal of the coin of the challenge but cannot get the related serial number from it ($\mathcal{S}$ fulfils the SA property). Moreover, $\mathcal{A}_{\mathtt{PA}_1^*}$ may have owned many times the coin of the challenge using the oracles `Spd`, `Rcv` and `Spd&Rcv`. But, by construction of $\mathcal{S}'$, all communications related to the spending of a coin are encrypted with a unilateral authenticated ephemeral session key. Thus, all communications of the final call to the `Spd&Rcv` oracle are encrypted, which means that

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

$\mathcal{A}_{\mathtt{PA}_1^*}$ cannot recognize the serial number of the spent coin embedded into the spending, except if $\mathcal{A}_{\mathtt{PA}_1^*}$ has succeeded in breaking either the security of KE or the security of $\mathcal{E}$ by decrypting the communications without knowing the decryption key. □

We then show that the previously defined anonymity properties and PA_1^* are independent (i.e. one property does not imply the other).

Proposition 1. *WA (resp. SA, resp. FA) and PA$_1$* are independent properties.*

Proof. WA $\not\Rightarrow$ PA$_1$*. The scheme proposed by Okamoto and Ohta [10, 11] fulfils the WA property but not the PA$_1$* one since the serial number of a coin is not protected and it is not modified from one spend to another.

PA$_1$* $\not\Rightarrow$ WA. If we apply the general construction of Section 4.1 onto a transferable e-cash scheme that does not fulfil the WA property, we can easily show that the new scheme fulfils PA$_1$* but not WA.

SA $\not\Rightarrow$ PA$_1$*. The schemes proposed in [14, 6, 5] fulfil the SA property but not the PA$_1$* one since the serial number of a coin is not protected and it does not change from one spend to another.

PA$_1$* $\not\Rightarrow$ SA. From SA$\Rightarrow$ WA and PA$_1$* $\not\Rightarrow$ WA, we have PA$_1$* $\not\Rightarrow$ SA.

FA $\not\Rightarrow$ PA$_1$*. See Appendix A.

PA$_1$* $\not\Rightarrow$ FA. This is due to FA$\Rightarrow$ SA and PA$_1$* $\not\Rightarrow$ SA. □

5.3 General description of a PA$_2$* scheme

In this section, we want to prove that PA$_2$* can be reached by a transferable e-cash scheme and the efficiency of the constructed scheme is out of the scope of this paper. The construction of a PA$_2$* transferable e-cash scheme is less straightforward than the PA$_1$*'s one. Indeed, we need to use an additional tool called *metaproof system* that has been introduced in [12] by de Santis and Yung.

Metaproofs. Roughly speaking, the metaproof tool corresponds to a NIZK (Non-Interactive Zero-Knowledge) proof of the existence of a NIZK proof to a statement. More precisely, they provide a metaproof system for 3SAT and prove that their metaproof system is a bounded NIZK proof system [1]. The metaproof system gives an *indirect* proof covered by additional encryption mechanism such that the metaprover possesses a zero-knowledge witness and does not necessary know the witness of the proof itself. Eventually, the metaproof can be applied recursively.

Overview of our PA$_2$* transferable e-cash scheme. A spent coin is classically represented by at least a serial number S, a security tag T (that permits the identification of double-spenders) and a proof that S and T are correct. Then, a transferable spent coin should consists in at least a serial number S, a set of security tags $\mathcal{T} = \{T_1, \cdots, T_l\}$ and a proof of validity.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

We first notice that, in a PA_2* e-cash scheme, a coin should be transferred without revealing any information on previous spends, even for the user that is receiving the coin. Moreover, the bank needs to retrieve the serial number and all security tags describing the history of the coin, which can be done by encrypting these values using the bank's public key.

Since a user should not be able to recognize a coin previously owned, the receiver should be able to verify the validity of the coin without being able to retrieve neither the serial number nor any security tag. Moreover, since the spender can next become a receiver of this coin, the spent coin should be modified at each spend so that she cannot recognize it. We thus need a cryptographic tool permitting someone to send the serial number, security tags and proofs without revealing nor knowing them but proving that they are valid, which can be done using metaproofs [12].

More precisely, if a user withdraws a coin, she spends it by computing the serial number S of the coin and the security tag T_1, plus a proof of validity V_1 that S and T_1 are well-formed. If the receiver wants to spend this coin, she computes a security tag T_2, she encrypts S and T_1 and she proves that T_2 is well-formed and that she knows the encryption of the serial number S, the encryption of the first security tag T_1 and a proof of validity V_1 without revealing the encrypted values nor V_1, using in particular a metaproof.

Description of our PA_2* Scheme. We largely use the proposal of transferable e-cash scheme from Canard, Gouget and Traoré [5] to describe our PA_2* scheme, with the restriction that a user withdraws one coin at a time, and not a wallet. In the following, we only give a high level description of our scheme and we refer to [5] and [12] for details.

Setup. Let $\mathcal{G}$ be a group of prime order p and g, g_0 be two random generators in $\mathcal{G}$. These data constitute the public parameters Par. Let $\mathcal{H}$ be a cryptographic hash function. In the `BKeyGen` algorithm, $\mathcal{B}$ computes two key pairs $(sk_{\mathcal{B},1}, pk_{\mathcal{B},1})$ and $(sk_{\mathcal{B},2}, pk_{\mathcal{B},2})$ of a CL signature scheme [3] that permit it to sign coins and enroll users, respectively. During the `UKeyGen` algorithm, each user $\mathcal{U}_i$ obtains a CL (verifiable) signature $C_i = \mathtt{Sign}(u_i, w_i)$ associated to his public key $pk_{\mathcal{U}_i} = g_0^{u_i}$ and a random data w_i. Let $\mathtt{Enc}_{\mathcal{B}}$ be a secure verifiable probabilistic encryption scheme (e.g. El Gamal) to encrypt messages for the bank.

Withdrawal protocol. Following [2, 5], a coin C withdrawn at the bank is a CL signature σ under the bank's public key $pk_{\mathcal{B},1}$ on the set of values (s, u_i, t, x) where u_i is the user secret key, s, t and x are random values; $C = (s, (u_i, t, x, \sigma))$. The value s implicitly defines the *serial number* of the coin and the value t implicitly defines the corresponding *security tag* (using the Dodis-Yampolskiy Pseudo Random Function [7]).

Spending of a withdrawn coin. A user $\mathcal{U}_i$, owning a coin $C = (s, (u_i, t, x, \sigma))$ withdrawn from $\mathcal{B}$, wants to spend a coin to a user $\mathcal{U}_j$.

1. $\mathcal{U}_j$ computes $r_j = g_0^{\frac{1}{u_j + d_j}}$ where d_j represents some data related to the transaction. Next, $\mathcal{U}_j$ sends r_j and d_j to $\mathcal{U}_i$.
2. $\mathcal{U}_i$ computes $S = g^s$, $T_i = pk_{\mathcal{U}_i} g^{\frac{r_j}{t + d_j}}$ and a NIZK proof V_i that
   - $\mathcal{U}_i$ knows a signature σ on s, u_i, t and x,

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

– $S = g^s$ and $T_i = pk_{\mathcal{U}_i} g^{\frac{r_j}{t+d_j}} = g_0^{u_i} g^{\frac{r_j}{t+d_j}}$,

without revealing s, t, u_i, x nor σ.

3. The spent coin is represented by $(S, \pi = (T_i, V_i, r_j, d_j))$.

First transfer of a coin. Let us now consider the user $\mathcal{U}_j$ that has received a coin $(S, \pi = (T_i, V_i, r_j, d_j))$ from user $\mathcal{U}_i$ during the above protocol. If $\mathcal{U}_j$ wants to spend this coin to a user $\mathcal{U}_k$, he has to proceed as follows.

1. $\mathcal{U}_k$ computes $r_k = g_0^{\frac{1}{u_k+d_k}}$ where d_k represents some data related to the transaction. Next, $\mathcal{U}_k$ sends r_k and d_k to $\mathcal{U}_j$.
2. $\mathcal{U}_j$ computes $T_j = pk_{\mathcal{U}_j} g^{\frac{r_k}{u_j+S+d_k}}$, $dS = \mathtt{Enc}_{\mathcal{B}}(S)$, $dT_i = \mathtt{Enc}_{\mathcal{B}}(T_i)$ and a NIZK proof V_j that
   – $\mathcal{U}_j$ knows a signature C_j on u_j and w_j
   – $T_j = pk_{\mathcal{U}_j} g^{\frac{r_k}{u_j+S+d_k}} = g_0^{u_i} g^{\frac{r_k}{u_j+S+d_k}}$ and $r_j = g_0^{\frac{1}{u_j+d_j}}$,
   – dS and dT_i are correct encryptions of the unrevealed values S and T_i, respectively,
   – there exists a NIZK proof V_i proving that S and T_i are well-formed, using in particular r_j and d_j, and linked to a valid signature of the bank (this step corresponds to a metaproof as described in [12]),

   without revealing u_j, S, C_j, w_j, r_j, d_j, T_i nor V_i.
3. The spent coin is represented by $(dS, \pi = (T_j, dT_i, V_j, r_k, d_k))$.

Second transfer of a coin. Let us now consider the user $\mathcal{U}_k$ that has received a coin $(dS, \pi = (T_j, dT_i, V_j, r_k, d_k))$ from user $\mathcal{U}_j$ during the above protocol. If $\mathcal{U}_k$ wants to spend it to $\mathcal{U}_l$, he has to proceed as follows.

1. $\mathcal{U}_l$ computes $r_l = g_0^{\frac{1}{u_l+d_l}}$ where d_l represents some data related to the transaction. Next, $\mathcal{U}_l$ sends r_l, d_l to $\mathcal{U}_k$.
2. $\mathcal{U}_k$ computes $T_k = pk_{\mathcal{U}_k} g^{\frac{r_l}{u_k+dS+d_l}}$, $d^2S = \mathtt{Enc}_{\mathcal{B}}(dS)$, $dT_j = \mathtt{Enc}_{\mathcal{B}}(T_j)$, and $d^2T_i = \mathtt{Enc}_{\mathcal{B}}(dT_i)$ and a NIZK proof V_k that
   – $\mathcal{U}_k$ knows a signature C_k on u_k and w_k
   – $T_k = pk_{\mathcal{U}_k} g^{\frac{r_l}{u_k+dS+d_l}} = g_0^{u_k} g^{\frac{r_l}{u_k+dS+d_l}}$ and $r_k = g_0^{\frac{1}{u_k+d_k}}$,
   – d^2S, dT_j and d^2T_i are correct encryption of the unrevealed values dS, T_j and dT_i, respectively,
   – there exists a NIZK proof V_j proving that dS, T_j and dT_i are well-formed, using in particular r_k and d_k, and linked to valid signatures of the bank, the one from the withdrawal phase and the one corresponding to the certificate of $\mathcal{U}_j$ (this step corresponds to a metaproof as described in [12]),

   without revealing u_k, dS, C_j, w_k, T_j, dT_i nor V_j.
3. The spent coin is represented by $(d^2S, \pi = (T_k, dT_j, d^2T_i, V_k, r_l, d_l))$.

The next spendings of this coin work similarly and are not described in this paper.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

Deposit and Identify. During a deposit, $\mathcal{U}$ sends the received coin (e.g. of the form $(d^2S, \pi = (T_k, dT_j, d^2T_i, V_k, r_l, d))$) to $\mathcal{B}$. Then $\mathcal{B}$ first checks if this coin is fresh by decrypting d^2S until obtaining the initial S and by testing if S already belongs to $\mathcal{L}$. If this is not the case, then everything is ok. Otherwise, there is a double-spending and $\mathcal{B}$ has two deposited coins. Then, $\mathcal{B}$ compares the first spending of both coins. If they are identical, then $\mathcal{B}$ goes to the second one an so on until two spends at the same level are different (this case always happens). $\mathcal{B}$ finally retrieves the identity of the cheater by first decrypting the related values T and T' and next using the compact e-cash technique to retrieve the cheater public key.

5.4 Achieving the PA$_2$* Property

Note that our PA$_2$* scheme is in accordance with the result of [6] which says that an unbounded adversary can always recognize his own coin during the game if it sees it later in a payment since such adversary is capable of decrypting values to retrieve the spender's identity.

Theorem 3. *Under the security of the used encryption scheme (e.g. El Gamal), the security of NIZK proofs and the security of the Dodis-Yampolskiy PRF, the proposed scheme fulfils the PA$_2$* property.*

Sketch of Proof. Assume that $\mathcal{A}_{\mathtt{PA}_2^*}$ succeeds in breaking the PA$_2$* property of the scheme described at Section 5.3. That means that $\mathcal{A}_{\mathtt{PA}_2^*}$ is able to decide, between two honest users i_0 and i_1 chosen by $\mathcal{A}_{\mathtt{PA}_2^*}$, which user is the spender i_b during a call to the oracle $\mathtt{Spd}(i_b)$. Note that, there is no restriction on the list of *authorized oracles* for such adversary.

The best strategy for $\mathcal{A}_{\mathtt{PA}_2^*}$ is to choose the users i_0 and i_1 such that he has previously manipulated all the coins owned by these users. Then $\mathcal{A}_{\mathtt{PA}_2^*}$ has to recognize the coin $C = (d^iS, \pi = (T_l, dT_k, \cdots, d^iT_j, V_l, r_m, d_m))$ sent by i_b that he has previously owned. Consequently, $\mathcal{A}_{\mathtt{PA}_2^*}$ knows some values that has been used to compute this coin (such as e.g. $d^{i_0}T_{j_0}$). When receiving a coin, $\mathcal{A}_{\mathtt{PA}_2^*}$ cannot learn anything from:

- the encrypted serial number d^iS under the security of the probabilistic encryption scheme (even if he knows the value that is encrypted),
- the encrypted security tags $dT_k, \cdots, d^iT_j$ under the security of the encryption scheme (even if he knows some encrypted values),
- the security tag T_l of the spender under the security of the Dodis-Yampolskiy PRF (see [7, 5] for details),
- the values r_m and d_m that comes from $\mathcal{A}_{\mathtt{PA}_2^*}$ himself,
- the proof V_l by definition of a NIZK proof. In particular, see the result on metaproofs [12] and on usual zero-knowledge proofs of knowledge based on the discrete logarithms [8, 4].

Consequently, even if $\mathcal{A}_{\mathtt{PA}_2^*}$ has already seen the spent coin, he cannot recognize it. Thus, $\mathcal{A}_{\mathtt{PA}_2^*}$ cannot win the PA$_2$* game and our construction is PA$_2$*, which concludes the proof. □

We finally show that there is no relation between previously defined anonymity properties and the PA$_2^*$ one.

Proposition 2. *PA$_2$* and WA (resp. SA, resp. FA, resp. PA$_1$*) are independent properties.*

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

Sketch of Proof. WA $\not\Rightarrow$ PA$_2$*. The scheme given in [10, 11] fulfils the WA property but not the PA$_2$* property since the serial number of a coin is not protected and it does not change from one spend to another.

PA$_2$* $\not\Rightarrow$ WA. See appendix B.

PA$_2$* $\not\Rightarrow$ SA. This is due to SA $\Rightarrow$ WA and PA$_2$* $\not\Rightarrow$ WA.

SA $\not\Rightarrow$ PA$_2$*. The schemes proposed in [14, 6, 5] fulfil the SA property but not the PA$_2$* property since the serial number of a coin is not protected and it does not change from one spend to another.

PA$_2$* $\not\Rightarrow$ FA. This comes from FA $\Rightarrow$ SA and PA$_2$* $\not\Rightarrow$ SA.

FA $\not\Rightarrow$ PA$_2$*. The scheme proposed in Section 4.1 fulfils the FA property but not the PA$_2$* property.

PA$_1$* $\not\Rightarrow$ PA$_2$*. The generic construction given in Section 4.1 fulfils PA$_1$* but not PA$_2$* property (for obvious reasons).

PA$_2$* $\not\Rightarrow$ PA$_1$*. The PA$_2$* scheme proposed in Section 5.3 does not fulfil the PA$_1$* property since the adversary being the bank in the PA$_1$* game can decrypt all encrypted data of all spends. □

6 Conclusion

In this paper, we provide the first study-in-depth of anonymity properties in transferable e-cash by introducing the full anonymity (FA) and perfect anonymity (PA). We show that the FA property can be reached by providing a generic construction and we prove that the PA cannot. We then define two restricted PA properties called PA$_1$* and PA$_2$* and we show that both restricted properties can be reached. Finally, we show that FA, PA$_1$* and PA$_2$* are three separate properties. Thus, an anonymous transferable e-cash scheme should ideally fulfils these three properties, which is the case (obviously inefficiently) if we apply the trick of Section 4.1 to the PA$_2$* scheme of Section 5.3. Note that all our results can easily be extended to wallets by using the compact e-cash techniques [2].

Acknowledgments. We are grateful to Marc Girault, Pascal Paillier and Jacques Traoré for their suggestions of improvement, and to anonymous referees for their valuable comments.

References

1. M. Blum, P. Feldman, and S. Micali. Non-Interactive Zero-Knowledge and Its Applications (Extended Abstract). In *STOC'88*, pages 103–112. ACM, 1988.
2. J. Camenisch, S. Hohenberger, and A. Lysyanskaya. Compact E-Cash. In *Eurocrypt'05*, volume 3494 of *LNCS*, pages 302–321. Springer, 2005.
3. J. Camenisch and A. Lysyanskaya. Signature Schemes and Anonymous Credentials from Bilinear Maps. In *Crypto'04*, volume 3152 of *LNCS*, pages 56–72. Springer, 2004.
4. S. Canard, I. Coisel, and J. Traoré. Complex Zero-Knowledge Proofs of Knowledge Are Easy to Use. In *ProvSec'07*, volume 4784 of *LNCS*, pages 122–137. Springer, 2007.
5. S. Canard, A. Gouget, and J. Traoré. Improvement of Efficiency in (Unconditional) Anonymous Transferable E-Cash. In *Financial Cryptography and Data Security'08*, volume 4887 of *LNCS*, pages 571–589. Springer, 2008.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

6. D. Chaum and T.P. Pedersen. Transferred Cash Grows in Size. In *Eurocrypt'92*, volume 658 of *LNCS*, pages 390–407. Springer, 1992.
7. Y. Dodis and A. Yampolskiy. A Verifiable Random Function with Short Proofs and Keys. In *PKC'05*, volume 3386 of *LNCS*, pages 416–431. Springer, 2005.
8. A. Kiayias, Y. Tsiounis, and M. Yung. Traceable Signatures. In *Eurocrypt'04*, volume 3027 of *LNCS*. Springer, 2004.
9. Y. Kim, A. Perrig, and G. Tsudik. Communication-Efficient Group Key Agreement. In *IFIP/Sec'01*, volume 193 of *IFIP Conference Proceedings*, pages 229–244. Kluwer, 2001.
10. T. Okamoto and K. Ohta. Disposable Zero-Knowledge Authentications and Their Applications to Untraceable Electronic Cash. In *Crypto'89*, volume 435 of *LNCS*, pages 481–496. Springer, 1990.
11. T. Okamoto and K. Ohta. Universal Electronic Cash. In *Crypto'91*, volume 547 of *LNCS*, pages 324–337. Springer, 1991.
12. A. De Santis and M. Yung. Cryptographic Applications of the Non-Interactive Metaproof and Many-Prover Systems. In *Crypto'90*, volume 537 of *LNCS*, pages 366–377. Springer, 1990.
13. M. Trolin. A Stronger Definition for Anonymous Electronic Cash. In *ePrint Archive*, 2006.
14. H. van Antwerpen. Electronic Cash. Master's thesis, CWI, 1990.

A Proof of Proposition 1: FA $\not\Rightarrow$ PA$_1$*

We first describe a toy scheme $\mathcal{TS}$ and next we prove that $\mathcal{TS}$ fulfils the FA property but it does not fulfil the PA$_1$* property.

Description of the Toy Scheme $\mathcal{TS}$. We assume that $\mathcal{S}$ is a transferable e-cash scheme that fulfils the SA property (e.g. [14, 6, 5]). We need to re-define only the spending protocol of $\mathcal{S}$. We additionally use as building blocks a secure symmetric encryption scheme $\mathcal{E} = (\texttt{Enc}, \texttt{Dec})$ and a unilateral authenticated group key agreement (GKA) scheme which uses e.g. the proposal of [9] where g is a public element. Each user has a signature key pair together with a valid certificate. In particular, this permits us to make the GKA scheme resistant to man-in-the-middle attacks. If $\mathcal{U}_1$ wants to spend a withdrawn coin to $\mathcal{U}_2$, he has to proceed as follows.

- $\mathcal{U}_1$ chooses at random a value K_1 and sends g^{K_1} to $\mathcal{U}_2$. User $\mathcal{U}_2$ chooses at random a value K_2, computes g^{K_2}, signs $g^{K_1} \| g^{K_2}$ and sends g^{K_2} and the signature to $\mathcal{U}_1$. Both can securely and secretly compute $K = g^{K_1 K_2}$ and execute a key-confirmation protocol.
- $\mathcal{U}_1$ and $\mathcal{U}_2$ play together the `Spend` protocol of $\mathcal{S}$ by encrypting communications using the encryption algorithm `Enc` with the common secret key K. Both $\mathcal{U}_1$ and $\mathcal{U}_2$ can decrypt communications using the decryption algorithm `Dec` with the common secret key K.

If $\mathcal{U}_2$ wants to spend a received coin to $\mathcal{U}_3$, the protocol is as follows.

- $\mathcal{U}_2$ sends $g^K = g^{g^{K_1 K_2}}$ to $\mathcal{U}_3$. User $\mathcal{U}_3$ chooses at random a value K_3, computes g^{K_3}, signs $g^{K_3} \| g^K$ and sends g^{K_3} and the signature to $\mathcal{U}_2$. Both can compute $K' = g^{K K_3} = g^{g^{K_1 K_2} K_3}$, as in [9], and execute a key-confirmation protocol.
- $\mathcal{U}_2$ and $\mathcal{U}_3$ play together the `Spend` protocol of $\mathcal{S}$ using `Enc` and the session key K', as for the spending of a withdrawn coin

Note that the adversary playing the role of $\mathcal{U}_2$ cannot take any advantage in not sending the correct value $g^K = g^{g^{K_1 K_2}}$ for obvious reasons.

Proposition 3. *Under the assumptions that $\mathcal{S}$ fulfils the SA property, and $\mathcal{E}$ is secure, the $\mathcal{TS}$ system fulfils the FA property.*

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

Sketch of Proof. Assume that $\mathcal{A}_{\mathtt{FA}}$ succeeds in breaking the FA property of $\mathcal{TS}$ and thus decide, between i_0 and i_1, which user is the user i_b from which $\mathcal{A}_{\mathtt{FA}}$ receives the coin of the challenge.

Note that the oracle `Rcv` is not allowed for the manipulation of users i_0 and i_1. Thus, at Step 2 of the Game, $\mathcal{A}_{\mathtt{FA}}$ chooses users i_0 and i_1 such that all the coins owned by users i_0 and i_1 have never been owned by $\mathcal{A}_{\mathtt{FA}}$. Then, $\mathcal{A}_{\mathtt{FA}}$ owns the coin of the challenge for the first time at step 4.

Before Step 3 of the game, $\mathcal{A}_{\mathtt{FA}}$ has observed the withdrawal of the coin of the challenge (using the oracle $\mathtt{With}(U)$) and $\mathcal{A}_{\mathtt{FA}}$ may have observed many times a spending between two honest users involving the coin of the challenge, using the oracles `Spd&Rcv`.

By assumption ($\mathcal{S}$ fulfils the SA property), $\mathcal{A}_{\mathtt{FA}}$ cannot get the serial number of a coin involved in a withdrawal protocol. By construction of $\mathcal{TS}$, all communications related to the spending of a coin are encrypted with an anonymous ephemeral session key. Thus, all communications related to a call to the oracle `Spd&Rcv` are encrypted. That means that $\mathcal{A}_{\mathtt{FA}}$ has no information about the identifier of the coin embedded into the spending ($\mathcal{A}_{\mathtt{FA}}$ may know the entry number of the coin in $\mathcal{OC}$ but not the serial number), except if $\mathcal{A}_{\mathtt{FA}}$ has succeeded in breaking either the security of the unilateral authenticated group key agreement or the security of $\mathcal{E}$ to decrypt the communications without knowing the decryption key which is impossible by assumption. □

Proposition 4. *$\mathcal{TS}$ does not fulfil the PA$_1$* property.*

Sketch of Proof. $\mathcal{TS}$ does not fulfil the PA$_1$* property (by construction). Indeed, $\mathcal{A}_{\mathtt{PA}_1^*}$ can always choose one of his coins and give it to i_0 that has no coin. Since $\mathcal{A}_{\mathtt{PA}_1^*}$ has received the coin, he necessarily knows the session key K. During the game, $\mathcal{A}_{\mathtt{PA}_1^*}$ chooses i_0 as defined previously and i_1 at random. Then, the oracle $\mathtt{Spd\&Rcv}(i_b, i)$, where i is a random honest user, is called. The underlying `Spend` protocol uses a session key K' from a key $\tilde{K}$ introduced by i_0 or i_1 and a random key K_3 introduced by the receiver, as in the spending protocol. Then $\mathcal{A}_{\mathtt{PA}_1^*}$ can easily check if the key $\tilde{K}$ corresponds or not to the key K he knows. If this is the case, $\mathcal{A}_{\mathtt{PA}_1^*}$ outputs 0 and he outputs 1 otherwise and wins the game with a probability of success equal to 1, which concludes the proof. □

B Proof of Proposition 2: PA$_2$* $\not\Rightarrow$ WA

In order to prove that PA$_2$* $\not\Rightarrow$ WA, we describe a toy scheme and we prove that it fulfils the PA$_2$* property but not the WA one.

Withdrawal protocol. The user $\mathcal{U}$ gets from the bank $\mathcal{B}$ a signature σ on the serial number s of the coin. Note that the serial number is not hidden to the bank that consequently knows s and σ.

Spending a withdrawn coin. The user $\mathcal{U}_1$ spends the coin (s, σ) to the user $\mathcal{U}_2$ by encrypting s and the signature σ to obtain E and producing a NIZK proof that the encrypted σ is a signature of the encrypted value s.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

Spending a received coin. $\mathcal{U}_2$ spends a received coin (E, U) to $\mathcal{U}_3$ by using the metaproof technique [12] to produce a NIZK proof V that there exists a NIZK proof U of validity of the spent coin. This step can be done many times so that the coin can be spent again and again.

This scheme is straightforwardly $\mathrm{PA}_2{}^*$ but does not achieve the WA property since the bank can decrypt all spends to retrieve s and thus make the link with the initial withdrawal.

Appeared in S. M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.): ACNS 2008, volume 5037 of LNCS, 2008.

Server-Aided Cryptography for Anonymity

Sébastien Canard and Iwen Coisel

Orange Labs R&D, 42 rue des Coutures, BP6243, F-14066 Caen Cedex, France

Abstract. Portable devices (mobile phones, smart cards, ...) are very useful to access services from anywhere. However, when authentication protocols require complex cryptography, implying costly mathematical operations, these devices may become inadequate because of their limited capabilities. This is in particular the case when the device must remain anonymous and unlinkable w.r.t. the service provider since it implies the use of complex cryptographic tools. In this paper we introduce the concept of *server-aided cryptography for anonymity* by adding a powerful intermediary which helps the restricted device in its cryptographic computations. We first give a general server-aided model in this setting, which model can be applied to several cryptographic tools: group, blind and ring signatures. We also provide secure and efficient server-aided variants of well-known constructions of these schemes, and prove that the proposed variants are the best one can obtain. To the best of our knowledge, this is the first complete study on server-aided cryptography for anonymous authentications.
Keywords. Server-aided, anonymous authentications, group signature, blind signature, ring signature.

1 Introduction

When operating in devices with restricted capabilities w.r.t. space memory and computing performances, cryptographic protocols sometimes need to be further studied. In practice, some cryptographic operations cannot be easily embedded in some devices such as SIM cards, smart cards or RFIDs. This is specially true when the number of computations to perform is important, which is the case when considering cryptographic tools for anonymous and unlinkable authentications, in which we focus in this paper.

There are several cryptographic tools that permit such anonymity: group signatures [1, 5, 23, 8, 34], ring signatures [43, 45, 16] and blind signatures [17, 36]. These primitives are moreover useful in some applications such as anonymous credentials, e-cash, e-vote and identity management. But they necessitate numerous modular exponentiations or pairing evaluations such that an implementation in a non-powerful device is not realistic. It is for example necessary to compute 11 modular exponentiations and 2 pairings to produce an XS group signature [23].

Considering that computers and smart cards become more and more powerful day after day, one possible strategy is to wait for sufficient capabilities before efficiently implementing such tools. But the need exists today, as shown by some commercial products (*e.g.* group signature based DAA in a TPM [47]), recent announcements (*e.g.* blind signature based Credentica patents bought by Microsoft) or some official recommendations (*e.g.* European Commission recommendation on the privacy in RFIDs [21]).

1.1 Related Work

The way to embed cryptography into non powerful devices has been largely studied by the cryptographic community. One solution is to make pre-computations. In fact, it is sometimes

possible to make some computations off-line and to store them, so that it is not necessary anymore to perform them on-line. This has the drawback of consuming a lot of space memory and thus simply shifts the problem. Another possibility is to modify the cryptographic mechanism to fit the device restrictions. This has already been done in the RFID case [31, 27] or when considering the integration of group and ring signatures in a smart card [14, 49]. This approach sometimes necessitates important modifications of the initial algorithm, and may imply some stronger (and questionable) assumptions such as *e.g.* the tamper-resistance one [14, 49].

In this paper, we focus on the approach which consists in providing the device some help to make the computations without any modification. A security model and some constructions exist for the verifier-side [32] in the context of the authentication process. The prover-side has been largely studied in the context of RSA [41, 2] where the aim is to help the restricted device to perform a modular exponentiation. Another approach has also been taken in the CAFE project [20, 22], which consists in designing schemes where a powerful prover interacts with a non-trusted smart card to perform some computations, and such that the prover is unlinkable *w.r.t.* the smart card.

In this paper, we study how a more powerful entity can help *e.g.* a smart card that is programmed to provide anonymous authentication, as done in [40] for the ACJT group signature scheme [1]. The same technique has already been used in the context of Trusted Platform Modules and the DAA schemes [9].

1.2 Our Contribution

In this paper, our main objective is to provide a general server-aided model where a restricted device is helped by an intermediary to compute anonymous and unlinkable transactions such as group or ring signatures. Considering these types of signatures, we stress that it is possible to modify most of (existing and future) constructions so that their efficiency be improved. In fact, in this paper, we focus on the case where the powerful intermediary intervenes between the prover and the verifier and we consider two practical cases. The first one is when the prover is the SIM card connected to the intermediary which role is played by the mobile phone. In this case, the SIM is already known by the mobile phone and thus, does not necessary want to be unlinkable *w.r.t.* this intermediary (we say that the latter is *trusted*). The second use case is when the prover is a contactless smart card and the intermediary is a publicly available card reader that can be reached by possibly many smart cards (we say that this intermediary is *untrusted*). For each existing construction, we also show that it is possible to describe the best secure server-aided variant, even if, for some example, this best computational gain is not very interesting. More precisely, we bring the following contribution to server-aided cryptography for anonymity.

1. We first give a generic server-aided model which permits a (possibly limited) prover to anonymously authenticate himself to a verifier. We introduce the role of the *intermediary* which helps the prover to perform the authentication. Our model takes into account group, blind and ring signature schemes. We first introduce the notion of *server-aided protocol*, next define the adversary and finally introduce security definitions. We thus introduce the notion of *server-aided soundness* which takes into account that the intermediary interacting with a prover may have access to some values specific to this prover that are not

publicly available (this is for example the case for some examples we give in the paper). After that, we focus on the *unlinkability property*. It may be then an easy task for the intermediary to link several authentications from one prover. We thus introduce two different types of unlinkability: the *weak* one which is used when the prover *trust* the intermediary and the *strong* one where the prover does not trust the intermediary. We finally define the *computation ratio*, which corresponds to the difference, in terms of efficiency, between the initial protocol and its server-aided version.

2. We next give a practical case for our generic server-aided model by using group signature schemes. We first focus on a generic construction of group signatures based on Camenisch-Lysyanskaya special signatures and secure in the random oracle model, which takes into account several existing schemes [1, 5, 23]. We present a weak-secure server-aided version and next propose a strong-secure server-aided version of the XS group system [23]. For these two proposals, we prove that this is the best secure variants one can obtain.
   For this purpose, we also give some words on server-aided zero-knowledge proofs of knowledge, as an important building block in this case. We finally give the server-aided version of a more recent scheme secure in the standard model and due to Groth [34].
3. We finally make a quite less complete study on server-aided blind and ring signature schemes. For all of them, we justify our generic model and next give the best secure server-aided variants of some existing practical schemes.

1.3 Organization of the Paper

After the present introduction, we first give in Section 2 some preliminaries that we will use all along the paper. We next introduce our general server-aided model in Section 3. After that, we give some words on server-aided zero-knowledge proofs of knowledge in Section 4 and next give a practical case for our generic server-aided model by using group signature schemes in Section 5.

2 Preliminaries

In this section, we give some useful tools and next focus on group signature schemes, which ones will illustrate our theoretical study all along the paper. We may have taken another type of scheme and some of them are given in the appendices.

2.1 Bilinear Groups

We first give some words on the bilinear maps, following [5]. Let G_1, G_2 and G_T be multiplicative cyclic groups of prime order p. g_1 (resp. g_2) is a generator of G_1 (resp. G_2). Finally, let e be a computable bilinear map $G_1 \times G_2 \longrightarrow G_T$ such that $e(g_1, g_2) \neq 1$ and for all $u \in G_1$, $v \in G_2$ and $a, b \in \mathbb{Z}$, $e(u^a, v^b) = e(u, v)^{ab}$. (G_1, G_2, G_T, e) is called a bilinear environment.

From the implementation point of view, it is today very difficult to compare the time efficiency of a pairing computation and the one of a modular exponentiation in a group of prime order. In [24], a detailed study of pairing implementations over Barreto-Naehrig curves is done by Devegili *et al.*. In a 32-bit 3.0 GHz Intel Pentium IV, they thus implement the Ate pairing in 23.2 ms and the Tate pairing in 33.8 ms. In a ADM 64 at 1.6 GHz, the computation

of one scalar multiplication in an elliptic curve (equivalent to the modular exponentiation) takes around 3 ms when the modulus is of size 1024 and the exponent is 160-bits long [25]. More generally, if we denote by E the time needed to perform one modular exponentiation (or a scalar multiplication in the elliptic curve setting) and by P the time needed to perform one pairing computation, the ratio between P and E is denoted by α, that is: $\alpha = \mathsf{P}/\mathsf{E}$. Using the above results [24, 25], the value α can for example be approximatively equal to 7 (for the Ate pairing).

2.2 Zero-Knowledge Proofs of Knowledge

Roughly speaking, a Zero Knowledge Proof of Knowledge (ZKPK), formalized by Feige, Fiat and Shamir in [26], is an interactive protocol during which an entity $\mathcal{P}$ proves to a verifier $\mathcal{V}$ that he knows a set of secret values $\alpha_1, \ldots, \alpha_q$ verifying a given relation R without revealing anything else. These protocols are also used to prove that some public values are well-formed from secret ones known by the prover.

In the sequel, we denote by $\textsc{Pok}(\alpha_1, \ldots, \alpha_q : \mathsf{R}(\alpha_1, \ldots, \alpha_q))$ a proof of knowledge of the secrets $\alpha_1, \ldots, \alpha_q$ verifying the relation R. We also define π as the interactive protocol between a prover $\mathcal{P}$, on input $\alpha_1, \ldots, \alpha_q$ and R and a verifier $\mathcal{V}$ on input R and which allows $\mathcal{P}$ to prove that she knows the secrets in a zero-knowledge manner. The output of $\mathcal{V}$ is either 1 if the prover is accepted and 0 otherwise.

2.3 Verifiable Encryption Schemes

We now introduce our notation for a verifiable public-key encryption scheme $\mathcal{E}$, which is composed of the procedures (KeyGen, Enc, Dec). The encryption public key is denoted epk and the corresponding decryption secret key is esk. We finally assume that there also exists a zero-knowledge proof of of knowledge of the plaintext of a revealed ciphertext denoted $\textsc{Pok}(\alpha : C = \textsc{Enc}(\alpha))$.

2.4 Signature Schemes with Additional Features

Camenisch and Lysyanskaya [11] have proposed various signature schemes to which they add some specific protocols:

- KeyGen is a probabilistic key generation algorithm which on input the security parameter 1^λ outputs the signature secret key ssk and the corresponding verification public key spk;
- an efficient protocol between a user (on input spk and the message $(x_1, \cdots, x_l)$) and a signer (on input ssk) that permits the user to obtain from the signer a signature σ of a Pedersen [42] commitment C on values $(x_1, \ldots, x_l)$ unknown from the signer. The latter computes $\sigma = \text{CL}\textsc{Sign}(C)$, which signature verifies the relation denoted $\sigma = \textsc{Sign}(x_1, \ldots, x_l)$;
- an efficient proof of knowledge of a signature of some committed values, denoted

$$\textsc{Pok}(\alpha_1, \ldots, \alpha_l, \beta : \beta = \textsc{Sign}(\alpha_1, \ldots, \alpha_l));$$

- Verif is a deterministic public algorithm which on input a message $(x_1, \cdots, x_l)$, a signature σ and spk outputs 1 if σ is a valid signature on the message, and 0 otherwise.

2.5 Group Signature Schemes

In [19], Chaum and van Heyst introduce the notion of *group signature* schemes where any member of the group can sign a document and any verifier can confirm that the signature has been computed by a group member. Moreover, group signatures are anonymous and unlinkable for every verifier except, when needed, for a given authority. Formally speaking [3], a group signature scheme $\mathcal{GS}$ is described by the following polynomial-time algorithms:

- GGen is a probabilistic algorithm which takes as input 1^λ and which outputs a tuple $(\mathsf{gpk}, \mathsf{gmsk}, \mathsf{osk})$ where gpk is the group's public key, gmsk is the secret key of the group manager and osk is the secret key of the opening manager;
- GJoin is a probabilistic protocol between the group manager and a new group member to provide the latter with her secret key $\mathsf{gsk}[i]$. The group manager makes an entry $\mathsf{reg}[i]$ in the registration table reg, with the entire transcript of the process.
- GSig is a probabilistic algorithm which takes as input a secret signing key $\mathsf{gsk}[i]$ and a message m and returns the group signature σ on m;
- GVer is a deterministic algorithm which takes as input the group public key gpk, a message m and a candidate signature σ and returns either 1 if the signature is valid or 0 otherwise;
- GOpen is a deterministic algorithm which takes as input the opening manager secret key osk, a message m and a signature σ and returns either an identity i or the symbol $\perp$ to indicate a failure, together with a proof τ of this claim;
- GJudge is a deterministic algorithm which takes as input the group public key gpk, a message m, a signature σ, an identity i and a proof τ and returns 1 if τ is a valid proof that user i has produce the signature σ and 0 otherwise.

3 The Server-Aided Model

We here give our generic server-aided model. We thus consider systems where there is a protocol between a prover and a verifier, such that the latter can anonymously authenticate the former. We introduce the notion of server-aided protocol in our setting, and next define the adversary and the security properties. We finally define a way to compare the efficiency of the initial and the server-aided version.

3.1 Notation and General Presentation

Let $\mathcal{P}$ (resp. $\mathcal{V}$, $\mathcal{I}$) be a probabilistic polynomial time Turing machine (PPTM) modeling the prover (resp. the verifier, the intermediary). We assume that there exists a system, denoted $\mathcal{S}$, with at least several provers, several verifiers and several intermediaries. We first assume that the key generation algorithm, denoted KeyGen of the system has already been executed and that the public parameters of the system, denoted pp, have already been generated, according to a security parameter λ. The prover $\mathcal{P}$ has access to a public key ppk and a corresponding secret key psk that may have been generated by interacting with a specific issuer (cf. GJoin for group signature schemes). We do not consider this phase in this paper since we only want to focus on the protocol between $\mathcal{P}$ and $\mathcal{V}$. Next, we consider π, which is a protocol between $\mathcal{P}$ and $\mathcal{V}$, with common input pp and specific input $(\mathsf{psk}, \mathsf{ppk}, \mathsf{in})$ for $\mathcal{P}$, which outputs some

values denoted out and halts by verifying a given predicate ϵ_π on input in and out. We denote $\epsilon_\pi = 1$ if the predicate is satisfied and 0 otherwise. At the end of the π protocol, the output of $\mathcal{P}$ (resp. $\mathcal{V}$) is denoted $\mathsf{view}^\pi_\mathcal{P}$ (resp. $\mathsf{view}^\pi_\mathcal{V}$). The scheme $\mathcal{S}$ may be composed of several other procedures but we only need the above ones for now. Some others will be given further when needed (see Section 5 and appendices). We now define the corresponding server-aided scheme, denoted $\mathcal{S}^*$. This scheme is similar to the $\mathcal{S}$ scheme, except that the π protocol is replaced by a server-aided one.

Definition 1 (Server-aided protocol). *Let π^* be a protocol between $\mathcal{P}$, $\mathcal{I}$ and $\mathcal{V}$. This protocol π^* is said* server-aided w.r.t. *a protocol π if and only if*

1. *$\mathcal{P}$, $\mathcal{I}$ and $\mathcal{V}$ have the common input pp and $\mathcal{P}$ has moreover a specific input $(\mathsf{psk}, \mathsf{ppk}, \mathsf{in})$;*
2. *the protocol π^* outputs the values out;*
3. *the protocol π^* halts by verifying the predicate ϵ_π.*

At the end of the π^* protocol, the output of $\mathcal{P}$ (resp. $\mathcal{V}$ and $\mathcal{I}$) is denoted $\mathsf{view}^{\pi^*}_\mathcal{P}$ (resp. $\mathsf{view}^{\pi^*}_\mathcal{V}$ and $\mathsf{view}^{\pi^*}_\mathcal{I}$). The protocol π^* is also denoted $\pi^*[\mathcal{P}(\mathsf{pp}, \mathsf{psk}, \mathsf{ppk}, \mathsf{in}), \mathcal{I}(\mathsf{pp}), \mathcal{V}(\mathsf{pp})]$.

Note that from one scheme $\mathcal{S}$ with a protocol π, it is possible to design several different variants of the corresponding server-aided protocol π^*. Our aim is to define the best one from both the security and the efficiency points of view.

Remark 1. In the following, we will consider that, in the server-aided protocol, there are no more communications between the prover and the verifier, i.e. all of them need to go through the intermediary $(\mathcal{P} \leftrightarrow \mathcal{I} \leftrightarrow \mathcal{V})$[1]. This is most of the time the case in real life since the prover is typically a smart card or a SIM card and the corresponding intermediary is the card reader, a PC or a mobile phone, while the verifier is an outside server or the network.

Focus on group signature schemes

We now focus on group signature schemes where the protocol π corresponds to the GSign protocol between a group member and a verifier. We consequently define the GSign* server-aided version of this protocol. The input in is the message m and the output out is the group signature σ. We next introduce the following definition concerning the corresponding predicate.

Definition 2 (Group signature predicate). *The predicate ϵ_π for group signature schemes is satisfied on input a message m and a signature σ if and only if* $\text{GVer}(\mathsf{gpk}, m, \sigma) = 1$.

3.2 Adversary and Oracles

We now define the adversary, denoted $\mathcal{A}$, as the entity which aims at breaking the security properties of the studied scheme ($\mathcal{S}$ or $\mathcal{S}^*$). As we focus on anonymous authentications, we have at least the three following properties for the $\mathcal{S}$ scheme: correctness, soundness and unlinkability. Again, this is the case for the studied schemes, such as group, blind and

[1] A discussion about $(\mathcal{I} \leftrightarrow \mathcal{P} \leftrightarrow \mathcal{V})$ is done below in Remark 2

ring signatures, even if, for some of them, there are some additional properties that will be described further in this paper, when needed.

In the following, we consider the scheme $\mathcal{S}$ as secure, that is it verifies all the above security properties. We now focus on the corresponding server-aided scheme $\mathcal{S}^*$. In this setting, we define $\mathcal{HP}$ as the set of honest provers and $\mathcal{CP}$ as the set of corrupted provers. Moreover, $\mathcal{A}$ may interact with some oracles that are defined as follows.

- AddP: by calling this add prover oracle with an input $\mathcal{P}$, the adversary introduces this new prover $\mathcal{P}$ in the system. The oracle adds $\mathcal{P}$ to the set $\mathcal{HP}$ of honest provers and assigns to this prover a new key pair $(\mathsf{ppk}, \mathsf{psk})$. The oracle outputs ppk.
- CrptP: this oracle takes as input a prover $\mathcal{P}$ and permits the adversary to obtain both psk and the total control on this prover $\mathcal{P}$ (that is, $\mathcal{A}$ has also the control of the random tape of the prover). The oracle moreover puts $\mathcal{P}$ in the set $\mathcal{CP}$ of corrupted provers.
- Reveal: this oracle, taking on input a prover $\mathcal{P}$, reveal the secret key psk of $\mathcal{P}$. Note that using this oracle, the adversary does not gain control of this user but simply obtains her secret key.
- Exec: the oracle plays alone all the π^* protocol on input a specific prover $\mathcal{P}$ and in. It outputs out.
- VProt: the oracle plays the role of both the prover and the intermediary during the π^* protocol. The adversary plays the role of the verifier.
- IProt: the oracle plays the role of the prover during the π^* protocol, while the adversary plays the role of the intermediary and the verifier.

In the following, we denote by $\mathcal{A}^{\text{Oracle}_1,\text{Oracle}_2}(\mathsf{in})$ the adversary taking on input in and having access to the oracles Oracle_1 and Oracle_2. $\mathcal{A}^*(\mathsf{in})$ means that the adversary $\mathcal{A}$ has access to all the above oracles. As we will see later, the fact that the adversary call the IProt with a particular prover implies that she can obtain from this prover, as an intermediary, some information that are not publicly available about this particular prover. We thus introduce the notion of weakly corrupted prover has follows.

Definition 3 (Weakly corrupted prover). *A prover $\mathcal{P}$ is said to be* weakly corrupted *during a given experiment if, at any time of this experiment, $\mathcal{A}$ call the* IProt *oracle on input $\mathcal{P}$. For a given experiment, the set $\mathcal{WCP}$ is the set of weakly corrupted provers.*

3.3 Correctness

Informally speaking, the scheme $\mathcal{S}$ is said to be *correct* if for a valid prover $\mathcal{P}$, the probability that a verifier $\mathcal{V}$ accepts $\mathcal{P}$ during an instance of the protocol π is overwhelming. In the server-aided setting, the correctness property needs to take into account that $\mathcal{I}$ participates in the new protocol π^*.

Each protocol π is, as described above, assigned to a predicate denoted ϵ_π. Here, we additionally introduce a *correctness predicate*, denoted $\epsilon_\pi^{\mathsf{corr}}$, which takes as input in and a prover $\mathcal{P}$ and which gives a description of the correctness conditions of a given scheme. One example of correctness predicate is given below for group signatures, and we also define this predicate for some other schemes in the appendices.

Focus on group signature schemes

We again focus on group signature schemes where the correctness needs to verify that the opening of a group signature truly rely to the correct member. We thus introduce the following definition.

Definition 4 (Group signature correctness predicate). *The correctness predicate $\epsilon_{\pi}^{\mathsf{corr}}$ for a group signature scheme is satisfied if and only if the three following conditions are verified for any message m and any group member i:*

- $\text{GVER}(\mathsf{gpk}, m, \text{GSIG}^*(i(\mathsf{gsk}[i], m), \mathcal{I}(\mathsf{gpk}), \mathcal{V}(\mathsf{gpk}))) = 1$;
- $\text{GOPEN}(\mathsf{osk}, m, \text{GSIG}^*(i(\mathsf{gsk}[i], m), \mathcal{I}(\mathsf{gpk}), \mathcal{V}(\mathsf{gpk}))) = (i, \tau)$ *and*
- $\text{GJUDGE}(\mathsf{gpk}, m, \text{GSIG}^*(i(\mathsf{gsk}[i], m), \mathcal{I}(\mathsf{gpk}), \mathcal{V}(\mathsf{gpk})), \tau) = 1$.

Using the correctness predicate, we now introduce the following experiment for the server-aided correctness property, where $\mathcal{A}$ is the adversary.

Server-aided correctness experiment $Exp_{\mathcal{S}^*,\mathcal{A}}^{\mathsf{SA-corr}}(\lambda)$:

- $\mathsf{pp} \longleftarrow \text{KEYGEN}(1^{\lambda})$; $\mathcal{HP} \longleftarrow \emptyset$; $\mathcal{CP} \longleftarrow \emptyset$;
- $(\mathsf{in}, \mathcal{P}) \longleftarrow \mathcal{A}^{\text{ADDP}}(\mathsf{pp})$;
- if $\mathcal{P} \notin \mathcal{HP}$, then return 0;
- $\mathsf{out} \longleftarrow \text{EXEC}(\mathcal{P}, \mathsf{in})$;
- return $\epsilon_{\pi}^{corr}(\mathsf{in}, \mathsf{out})$;

For a server-aided system $\mathcal{S}^*$, we define the probability of success of an adversary $\mathcal{A}$ for the SA correctness experiment as follows:

$$Succ_{\mathcal{S}^*,\mathcal{A}}^{\mathsf{SA-corr}}(\lambda) = \Pr\left[Exp_{\mathcal{S}^*,\mathcal{A}}^{\mathsf{SA-corr}}(\lambda) = 1\right].$$

Definition 5 (SA-correctness). *A scheme $\mathcal{S}^*$ is* SA-correct *if for any polynomial-time adversary $\mathcal{A}$, $Succ_{\mathcal{S}^*,\mathcal{A}}^{\mathsf{SA-corr}}(\cdot)$ is negligible.*

Remark 2. As $\mathcal{I}$ makes the link between $\mathcal{P}$ and $\mathcal{I}$, there is no way to prevent her from sending false values to $\mathcal{V}$ so that a valid $\mathcal{P}$ is rejected. In case $\mathcal{I}$ is behind $\mathcal{P}$ and has no link with $\mathcal{V}$, it is possible to define a strong cooperative completeness where for a valid $\mathcal{P}$ interacting with a corrupted $\tilde{\mathcal{I}}$, the probability that $\mathcal{V}$ π^*-accepts $\mathcal{P}$ during an instance of the protocol π^* is overwhelming. Since this is not the practical case we are studying, we will not consider this case in the rest of the paper. Note that one possibility can be for $\mathcal{V}$ to output either $\epsilon_{\pi^*} = 1$ if the predicate is verified, $\epsilon_{\pi^*} = 0$ if she is sure that this is not a true prover and $\epsilon_{\pi^*} = \perp$ if she is not sure and thus needs to ask $\mathcal{P}$ to avoid any ambiguity.

3.4 Soundness

We define a scheme $\mathcal{S}$ as *sound* if for a fake $\tilde{\mathcal{P}}$, the probability that $\mathcal{V}$ accepts $\tilde{\mathcal{P}}$ during an instance of π is negligible. More generally, the adversary has to produce a valid authentication with some precise properties, depending on the properties of the studied scheme. As a

consequence, we additionally introduce a *soundness predicate*, denoted $\epsilon_{\pi}^{\mathsf{sound}}$, which takes as input in and out and which gives a description of the security soundness of a given scheme $\mathcal{S}$. This predicate is satisfied ($\epsilon_{\pi}^{\mathsf{sound}} = 1$) if $\mathcal{A}$ is successful and is not satisfied ($\epsilon_{\pi}^{\mathsf{sound}} = 0$) otherwise. One example of soundness predicate is given below for group signatures (see also the appendices).

Focus on group signature schemes

We again focus on the group signature scheme case where the soundness property should include the fact that an adversary is not able to produce a valid signature which can not be linked to her using the GOpen and GJudge procedures. We thus introduce the following definition (see [3] for details).

Definition 6 (Group signature soundness predicate). *The soundness predicate ϵ_{π}^{sound} for a group signature scheme is satisfied if and only if, on input m and σ, all the following conditions are verified:*

- $\text{GVer}(gpk, m, \sigma) = 1$;
- $\text{GOpen}(osk, m, \sigma) = \perp$ *or* $\text{GJudge}(gpk, m, \sigma, i, \tau) = 0$ *where* $(i, \tau) = \text{GOpen}(osk, m, \sigma)$.

In the server-aided setting, the new soundness property should take into account the participation of $\mathcal{I}$ in π^*. As $\mathcal{I}$ interacts with $\mathcal{P}$ so as to help her, $\mathcal{I}$ may have access to some values that are not publicly available. The adversary can then easily impersonate $\mathcal{I}$ during the soundness experiment by doing the same (correct) computations as an honest $\mathcal{I}$ does. So, it is obvious that $\mathcal{A}$ should play the role of both $\mathcal{P}$ and $\mathcal{I}$. $\mathcal{A}$ consequently interacts with a verifier $\mathcal{V}$ and her aim is to output $(\mathsf{in}, \mathsf{out})$ such that $\epsilon_{\pi}^{\mathsf{sound}}(\mathsf{in}, \mathsf{out}) = 1$. More formally, we define the following experiment where $\mathcal{A}$ is the adversary.

Server-aided soundness experiment $Exp_{\mathcal{S}^*,\mathcal{A}}^{\mathsf{SA-sound}}(\lambda)$:

- $\mathsf{pp} \longleftarrow \text{KeyGen}(1^{\lambda})$; $\mathcal{HP} \longleftarrow \emptyset$; $\mathcal{CP} \longleftarrow \emptyset$; $\mathcal{WCP} \longleftarrow \emptyset$;
- $(\mathsf{in}, \mathsf{out}) \longleftarrow \mathcal{A}^*(\mathsf{pp})$;
- return $\epsilon_{\pi}^{\mathsf{sound}}(\mathsf{in}, \mathsf{out})$;

For a system $\mathcal{S}^*$, we define the probability of success of $\mathcal{A}$ for the SA soundness experiment as follows:

$$Succ_{\mathcal{S}^*,\mathcal{A}}^{\mathsf{SA-sound}}(\lambda) = \Pr\left[Exp_{\mathcal{S}^*,\mathcal{A}}^{\mathsf{SA-sound}}(\lambda) = 1\right].$$

Definition 7 (SA-soundness). *A scheme $\mathcal{S}^*$ is* SA-sound *if for any polynomial-time adversary $\mathcal{A}$, $Succ_{\mathcal{S}^*,\mathcal{A}}^{SA-sound}(\cdot)$ is negligible.*

Lemma 1. *Let $\mathcal{S}$ be a scheme and $\mathcal{S}^*$ be the corresponding server-aided scheme. If $\mathcal{S}^*$ is not SA-sound while the set $\mathcal{WCP}$ is empty at the end of the soundness experiment, then $\mathcal{S}$ is not sound.*

Remark 3. An honest intermediary $\mathcal{I}$ may know that she is interacting with the adversary $\mathcal{A}$ but we can note that this is just an efficiency issue. In fact, if $\mathcal{I}$ sees that $\mathcal{A}$ does not know

the secrets, she will interrupt the protocol π^* so that $\mathcal{V}$ needs not to verify the predicate ϵ_π. However, $\mathcal{A}$ may falsely accuse $\mathcal{I}$ but he will not be able to prove that she is an honest prover after being accused of.

3.5 Unlinkability

In this paper, we mainly focus on the *unlinkability* property since it typically implies costly mathematical operations and, consequently, our server-aided model is more useful in this case. Informally, a scheme $\mathcal{S}$ is said to be *unlinkable* if an adversary cannot distinguish between two honest provers, which one is playing an instance of π with him. In the server-aided setting, the adversary may be able to interact several times with the same prover and thus learn some specific values. Then, it may be an easy task to link several authentications from one prover, breaking the unlinkability property. However, this is not so restricting since the prover may be in practice a SIM card connected to a trusted[2] mobile phone. Moreover, as we will see further in Section 5, the computational ratio may be more attractive in this case. In the following, we thus consider two different types of unlinkability.

Weak server-aided unlinkability. The weak version of the server-aided unlinkability assumes that the prover trusts the intermediary and, as a consequence, that $\mathcal{P}$ does not necessary want to be anonymous *w.r.t.* $\mathcal{I}$. More formally, we define the following experiment, where $b \in \{0,1\}$ is a bit associated to the experiment and $\mathcal{A}$ is the adversary.

Server-aided weak unlinkability experiment $Exp^{\mathsf{weak-unlink}-b}_{\mathcal{S}^*,\mathcal{A}}(\lambda)$:

- $\mathsf{pp} \longleftarrow \textsc{KeyGen}(1^\lambda)$; $\mathcal{HP} \longleftarrow \emptyset$; $\mathcal{CP} \longleftarrow \emptyset$; $\mathcal{WCP} \longleftarrow \emptyset$;
- $(\mathcal{P}_0, \mathcal{P}_1, \mathsf{in}) \longleftarrow \mathcal{A}^*(\mathsf{pp})$
- $\mathsf{out} \longleftarrow \textsc{VProt}(\mathcal{P}_b, \mathsf{in})$
- $b' \longleftarrow \mathcal{A}^*(\mathsf{in}, \mathsf{out})$

For a scheme $\mathcal{S}^*$, we define the advantage of $\mathcal{A}$ for the weak unlinkability experiment as follows:

$$Adv^{\mathsf{weak-unlink}}_{\mathcal{S}^*,\mathcal{A}}(\lambda) = \Pr\left[Exp^{\mathsf{weak-unlink}-1}_{\mathcal{S}^*,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[Exp^{\mathsf{weak-unlink}-0}_{S^*,\mathcal{A}}(\lambda) = 1\right].$$

Definition 8 (SA-weak-unlinkability). *A scheme $\mathcal{S}^*$ is* SA-weak-unlinkable *if for any polynomial-time adversary $\mathcal{A}$, $Adv^{\mathsf{weak-unlink}}_{\mathcal{S}^*,\mathcal{A}}(\cdot)$ is negligible.*

Lemma 2. *Let $\mathcal{S}$ be a scheme and $\mathcal{S}^*$ be the corresponding server-aided scheme. $\mathcal{S}^*$ is SA-weak-unlinkable if and only if $\mathcal{S}$ is unlinkable.*

Proof (sketch). In both experiments (initial and weak ones), the role of the adversary is the same during the focused protocol. Moreover, all the information the adversary may have obtained using the VProt oracle can also be obtained using the Reveal one, which concludes the proof. □

[2] Here, trusted is related to the anonymity property. It consequently implies that the smart card (resp. SIM card) is not anonymous w.r.t. the card reader (resp. the mobile phone).

Strong server-aided unlinkability. In some cases, being unlinkable *w.r.t.* the intermediary is a really important issue and needs to be studied. We consequently provide a stronger definition for the server-aided unlinkability property, where the main difference with the previous one is that $\mathcal{P}$ now does not trust $\mathcal{I}$ and wants to be anonymous and unlinkable also *w.r.t.* $\mathcal{I}$. In the following experiment, the adversary now plays the role of the intermediary during the focused π^* protocol. More formally, we define the following experiment, where $b \in \{0,1\}$ is a bit associated to the experiment and $\mathcal{A}$ is the adversary.

Server-aided strong unlinkability experiment $Exp^{\mathsf{strong-unlink}-b}_{\mathcal{S}^*,\mathcal{A}}(\lambda)$:

- $\mathsf{pp} \longleftarrow \textsc{KeyGen}(1^\lambda)$; $\mathcal{HP} \longleftarrow \emptyset$; $\mathcal{CP} \longleftarrow \emptyset$; $\mathcal{WCP} \longleftarrow \emptyset$;
- $(\mathcal{P}_0, \mathcal{P}_1, \mathsf{in}) \longleftarrow \mathcal{A}^*(\mathsf{pp})$
- $\mathsf{out} \longleftarrow \textsc{IProt}(\mathcal{P}_b, \mathsf{in})$
- $b' \longleftarrow \mathcal{A}^*(\mathsf{in}, \mathsf{out})$

For a scheme $\mathcal{S}^*$, we define the advantage of $\mathcal{A}$ for the strong unlinkability experiment as follows:

$$Adv^{\mathsf{strong-unlink}}_{\mathcal{S}^*,\mathcal{A}}(\lambda) = \Pr\left[Exp^{\mathsf{strong-unlink}-1}_{\mathcal{S}^*,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[Exp^{\mathsf{strong-unlink}-0}_{\mathcal{S}^*,\mathcal{A}}(\lambda) = 1\right].$$

Definition 9 (SA-strong-unlinkability). *A scheme $\mathcal{S}^*$ is* SA-strong-unlinkable *if for any polynomial-time adversary $\mathcal{A}$, $Adv^{\mathsf{strong-unlink}}_{\mathcal{S}^*,\mathcal{A}}(\cdot)$ is negligible.*

Definition 10 (Secure server-aided scheme).
$\mathcal{S}^*$ is server-aided secure *if and only if $\mathcal{S}^*$ is SA-correct, SA-sound and SA-strong-unlinkable.*

$\mathcal{S}^*$ is server-aided weak secure *if and only if $\mathcal{S}^*$ is SA-correct, SA-sound and SA-weak-unlinkable.*

3.6 Computational Ratio

The aim of server-aided cryptography is to decrease the computational complexity. Thus, there is no interest if the server-aided protocol π^* is less efficient than the standard one π. We consequently need to measure this efficiency for both protocols. For this purpose, we first introduce the following definition.

Definition 11 (Computational ratio).
Let κ be the computational cost of $\mathcal{P}$ during the execution of the π protocol.

Let κ^* be the computational cost of $\mathcal{P}$ during the execution of the server-aided protocol π^*.

The computational ratio *γ of the server-aided protocol is equal to κ^*/κ.*

It is obvious that the computational ratio lies in $[0,1]$ and that it should be as smallest as possible to obtain a better efficiency gain. As said before, for one specific π protocol, it is possible to design several variants of the server-aided protocol π^*. Our aim is thus to design the most efficient and secure variant of π^*. We consequently introduce the following definition.

Definition 12 (Best computational ratio).
Let Γ be the computational ratio of one secure variant of the π^* protocol. If all other variants of π^* with computational ratio $\gamma < \Gamma$ are not secure, then Γ is called the best computational ratio *for π^*.*

The variant of the π^* protocol with a computational ratio Γ is called the best server-aided variant*.*

Remark 4. It is possible to refine this comparison by taking into account the computational complexity of $\mathcal{I}$. In fact, we can compare the complexity of the initial protocol with the one of the server-aided variant, taking into account the computational complexity of $\mathcal{P}$ plus the one of $\mathcal{I}$. Note also that $\mathcal{I}$ and $\mathcal{P}$ may work on parallel so as to make a better improvement on the global time complexity.

In Section 5, we will apply our model and methodology to group signature schemes, by giving some practical SA versions of existing schemes and proving that they are the best ones that can be achieved. Before focusing on group signature schemes, we need first to give some words on zero-knowledge proofs of knowledge, which is an important building block for our constructions in Section 5.

4 The Case of ZKPK

As said in the preliminaries, a ZKPK is an interactive protocol during which one entity proves to a verifier, in a zero-knowledge, that she knows some secrets verifying a given relation R. In the following, we will need, in some cases, that a zero-knowledge proof of knowledge is done in a server-aided way. Our aim is next to define such protocol without compromising the security of the ZKPK.

Most of existing honest verifier ZKPK[3] are iterations of three-move protocols between $\mathcal{P}$ and $\mathcal{V}$. In the first step ($\mathcal{P} \longrightarrow \mathcal{V}$), the prover computes the commitment by replacing each secret value in R by a related randomly chosen value r. This step typically implies modular exponentiations and is the most expensive part of the ZKPK. In the second step, the challenge one, $\mathcal{V}$ chooses a random challenge and sends it to $\mathcal{P}$. In the last one $\mathcal{P}$ sends to $\mathcal{V}$ the responses using the random values of the commitment, the challenge and the secrets. Note that the knowledge of r is equivalent to the knowledge of the secret. As a consequence, if the intermediary knows all the randomness used in the challenge, she also knows the related secrets. As a consequence, either $\mathcal{I}$ is on charge of one secret and in this case, he can compute the challenge and the response, or $\mathcal{I}$ can not do anything related to this secret.

More precisely, let us consider $\textsc{Pok}(\alpha_1, \ldots, \alpha_q : \mathsf{R}(\alpha_1, \ldots, \alpha_q))$, where $\alpha_1, \ldots, \alpha_q$ are the secrets and R is the relation verified by the secrets (see Section 4.1 below for examples of relation). We want to design a ZKPK protocol such that the the prover is on charge of the n first secrets $\alpha_1, \ldots, \alpha_n$ and such that the intermediary is on charge of the $q - n$ other ones $\alpha_{n+1}, \ldots, \alpha_q$. The choice of these values needs to be carefully done so that looked

[3] Note that as we consider the anonymity context, we will only consider so-called *honest verifier zero-knowledge* since this is the corresponding suitable setting. The advantage is that one iteration of three-move protocols may suffice.

information is not sufficient for the intermediary to break soundness of the system. Moreover, the protocol should be designed such that $\alpha_1, \ldots, \alpha_n$ are not compromised. We denote by $\text{SAPoK}(\mathcal{P}(\alpha_1, \ldots, \alpha_n), \mathcal{I}(\alpha_{n+1}, \ldots, \alpha_q) : \mathsf{R}(\alpha_1, \ldots, \alpha_q))$ the server-aided ZKPK done this way.

4.1 Discrete Logarithm Relation Sets

Most of the time, the studied cases consider secrets as discrete logarithms in relations constructed over a group G either of prime or unknown order. These relations are typically the following ones: proof of knowledge of a discrete logarithm $\text{PoK}(\alpha : y = g^\alpha)$ [44, 33]; proof of knowledge of a representation $\text{PoK}(\alpha_1, \ldots, \alpha_q : y = g_1^{\alpha_1} \ldots g_q^{\alpha_q})$ [30]; proof of equality of discrete logarithms $\text{PoK}(\alpha : y = g^\alpha \wedge z = h^\alpha)$ [18] and proof that a committed value lies in a given interval I $\text{PoK}(\alpha : y = g^\alpha \wedge \alpha \in I)$ [15, 7, 10].

Any relation R that is a combination of the 4 above types of proofs of knowledge is sometimes called a Discrete Logarithm Relation Set [37] (denoted DLRS in the sequel). In this case, it is possible to design a generic protocol [37] which is a secure ZKPK, as stated in [13] and refined in [12].

Definition 13. *Let G be a cyclic group of known or unknown order. A relation R with z equations over a set of q secret variables $\{\alpha_1, \ldots, \alpha_q\}$ and $q+1$ objects per relation $A_{i,1}, \ldots, A_{i,q}, T_i \in G$ (for the i-th relation) is a DLRS if (i) each object $A_{i,w}$ is related to the secret variable α_w in the i-th relation (equal to 1 if α_w is not implied in the i-th relation); (ii) the i-th equation verifies $\prod_{w=1}^{q} A_{i,\omega}^{\alpha_\omega} = T_i$ and (iii) every free variable α_ω is assumed to take values in a finite integer range $]2^{l_\omega} - 2^{\mu_\omega}, 2^{l_\omega} + 2^{\mu_\omega}[$ where $l_\omega, \mu_\omega \geq 0$. The underlying* PoK *is denoted* $\text{PoK}(\alpha_1, \ldots, \alpha_q : \prod_{\omega=1}^{q} A_{1,\omega}^{\alpha_\omega} = T_1 \wedge \ldots \wedge \prod_{\omega=1}^{q} A_{z,\omega}^{\alpha_\omega} = T_z)$.

Focus on the XS signature of knowledge [23]

We now focus on the XS group signature scheme [23], where a group signature includes the ZKPK $\text{PoK}\big(z, \alpha, \beta, u : T_1 = k^\alpha \wedge T_3 = k^\beta \wedge T_2/T_4 = h^\alpha/g^\beta \wedge e(T_2, g_2)^u e_{hw}^{-\alpha} e_{hg}^{-z} = e_{gg}/e(T_2, w)\big)(m)$, which is a DLRS with 4 relations, 4 secret variables (z, α, β, u), and 11 objects in total.

In case of a group of unknown order, we denote by $\mathcal{R}_w$ the set $\pm\{0,1\}^{\eta(\mu_w + k)}$ where η and k are security parameters [37, 13]. If we are in a group of prime order p, $\mathcal{R}_w$ is then $\mathbb{Z}_p$ and the computations of the s_w's are performed modulo p. Let $\mathcal{Z}$ be the set $[1, z]$, and for all $i \in \mathcal{Z}$, $\mathcal{W}_i = \{w \in [1, q] : A_{i,w} \neq 1\}$, $\mathcal{W}_{i,\mathcal{P}} = \mathcal{W}_i \cap [1, n]$ and $\mathcal{W}_{i,\mathcal{I}} = \mathcal{W}_i \cap [n+1, q]$. The standard version and the verification equation can be found in [37, 13, 12] and is not detailed in this paper, due to space limitations. We next detail in Figure 1 the server-aided version of a DLRS protocol between the prover (taking on input $\alpha_1, \ldots, \alpha_q$ and the DLRS relation R), the intermediary (taking on input $\alpha_{n+1}, \ldots, \alpha_q$ and R) and the verifier (on input R).

Remark 5. We here consider that $\mathcal{P}$ makes herself the multiplications during the commitment phase. Note that it does not matter, from the security point of view, if these multiplications are done by $\mathcal{P}$ or $\mathcal{I}$.

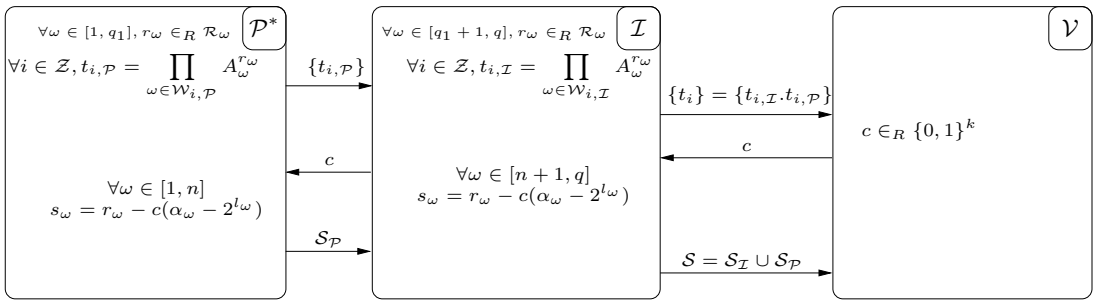

Fig. 1. server-Aided Discrete-log Relation Set R

Security Considerations. In fact, if we consider the protocol between $\mathcal{P}$ and $\mathcal{I}$ in Figure 1, we remark that they run the protocol of the ZKPK $\textsc{Pok}(\alpha_1, \ldots, \alpha_n : \prod_{\omega \in \mathcal{W}_{1,\mathcal{P}}} A_{1,\omega}^{\alpha_\omega} = T_1 \wedge \ldots \wedge \prod_{\omega \in \mathcal{W}_{z,\mathcal{P}}}^{n} A_{z,\omega}^{\alpha_\omega} = T_z)$, which has been proved to be secure in [13, 12]. As a consequence, the soundness of the global protocol is preserved since the intermediary is not able to perform the global ZKPK without the prover. The zero-knowledge property is also maintained for $\alpha_1, \cdots, \alpha_n$ but not for the other secrets.

4.2 Non-Interactive Proofs

ZKPK can also be done non-interactively and in this case, we refer to signature of knowledge (Sok and SASok). One method to convert an interactive protocol into a non-interactive one consists in using the Fiat-Shamir heuristic [28], in the random oracle model. The prover computes a pseudo-random challenge c by using a hash function $\mathcal{H} : \{0,1\}^* \to \{0,1\}^k$ taking on input the concatenation of the public parameters, the commitments and optionally an additional message m. The verification step consists then in testing whether or not c has been correctly computed by $\mathcal{P}$. In the server-aided setting, this step may be done by $\mathcal{I}$ since a bad challenge does not imply any security flaw. The rest of the computations are unchanged.

5 The Case of Group Signature Schemes

Our aim is now to give a practical case for our model defined above and next to give practical instances of GSign for some schemes. We first focus on a generic construction of group signatures based on CL signatures and secure in the random oracle model. Security proofs are moved to Appendix A. Note that in the BSZ group signature scheme model [3], there is an additional security property called non-frameability where $\mathcal{A}$ should not create a proof that an honest user produced a certain valid signature. The introduction of an intermediary in the SA model does not modify this property, which can be find in [3].

We also produces two different versions of the XSGS group signature scheme [23], one with weak security and one with strong security. Even if these variants can be viewed as trivial or useless in terms of efficiency, our aim here is to describe the best variants that can be obtained from this group signature scheme.

5.1 A Generic (Server-Aided) Construction

It is possible to design a group signature scheme $\mathcal{GS}$ using a CL special signature scheme (see [11] and Section 2) and an encryption scheme. The group signature scheme described in [23] is one example, but there are other ones [1, 5]. They are all based on the following construction[4].

- GGen: this phase executes the key generation algorithm of both the CL special signature (for the join procedure) and the encryption schemes (for the anonymity revocation).
- GJoin: this protocol is based on the CL signing protocol between the group manager (acting as the signer) and the user (that is, the new group member). At the end of the protocol, the group member obtains a signature σ of a secret value denoted usk. The value σ corresponds to one or several group elements $(A_1, \cdots, A_\ell)$ and/or one or several exponents $(e_1, \cdots, e_k)$.
- GSig: this procedure is executed by the group member, on input a message m. She first produces the encryption of *e.g.* A_1 of the signature σ, that is $C = \textsc{Enc}(A_1)$, and next the signature of knowledge (using the Fiat-Shamir heuristic) $U = \textsc{Sok}(A_1, \cdots, A_\ell, e_1, \cdots, e_k, x : (A_1, \cdots, A_\ell, e_1, \cdots, e_k) = \textsc{Sign}(x) \wedge C = \textsc{Enc}(A_1))(m)$. The group signature is finally (C, U)
- GVer: this step only consists in verifying the signature of knowledge U.
- GOpen: the opening manager, on input the decryption key, can decrypt C to retrieve A_1. She also produces a ZKPK, verified by GJudge, that she has correctly decrypted the signature.

Weak-secure variant. We now focus on a first variant of the group signature procedure GSign which is weak-secure, and denoted $\mathcal{GS}^*_{\mathsf{weak}}$. In the new protocol, the group member first sends to the intermediary $\mathcal{I}$ the CL signature she has obtained in the join protocol, that is $(A_1, \cdots, A_\ell, e_1, \cdots, e_k)$. The intermediary next produces the encryption $C = \textsc{Enc}(A_1)$ of the value A_1. Finally, as the proof of knowledge U defined above is most of the time a DLRS relation, the group member and $\mathcal{I}$ next engage in a server-aided signature of knowledge protocol such as described in Section 4. This proof is denoted

$$U = \text{SASok}(\mathcal{P}(x), \mathcal{I}(A_1, \cdots, A_\ell, e_1, \cdots, e_k) : (A_1, \cdots, A_\ell, e_1, \cdots, e_k) = \textsc{Sign}(x) \wedge C = \textsc{Enc}(A_1))(m).$$

Theorem 1. *$\mathcal{GS}^*_{\mathsf{weak}}$ is the best server-aided weak variant of $\mathcal{GS}$.*

[4] Note that we do not give the details of the Join procedure since this is not our aim in this paper to improve this phase.

Focus on the XS group signature scheme [23]

We now focus on the XS group signature scheme [23]. Let (G_1, G_2, G_T, e) be a bilinear environment.

CL signature scheme. In this case, this is the one due to Boneh *et al.* [5]. The secret key ssk is a random γ and the public key is $w = g_2^\gamma$, where $g_2 \in G_2$. The signature of a message x is the couple (A, u) such that $A^{u+\gamma} = g_1 h^x$ where g_1 and h are random generators of the group G_1.

Encryption scheme. The one chosen in [23] is the double El Gamal encryption scheme. In this case, $h = k^{\zeta_1}$ and $g = k^{\zeta_2}$ are the encryption public keys where $\zeta_1, \zeta_2 \in \mathbb{Z}_p^*$ are the secret keys and $k \in G_1$. The encryption of A is done by computing $T_1 = k^\alpha$, $T_2 = Ah^\alpha$, $T_3 = k^\beta$, $T_4 = Ag^\beta$ where $\alpha, \beta \in \mathbb{Z}_p^*$.

Signature of knowledge. We finally obtain the following signature of knowledge:

$$U = \text{SASOK}\big(\mathcal{P}(z), \mathcal{I}(\alpha, \beta, u) : T_1 = k^\alpha \wedge T_3 = k^\beta \wedge T_2/T_4 = h^\alpha/g^\beta \wedge e(T_2, g_2)^u e_{hw}^{-\alpha} e_{hg}^{-z} = e_{gg}/e(T_2, w)\big)(m)$$

where $e_{hg} = e(h, g_2)$ and $e_{gg} = e(g_1, g_2)$ are public and $z = u\alpha + x$, which is done as described in Figure 1. The signature on the message m is finally $\sigma = (T_1, T_2, T_3, T_4, U)$.

Computational gain. If we take the experiment measures given in Section 2.1, the computation gain for this server-aided variant is $\frac{1}{11+2\alpha}$ and thus $1/25 = 0.04$ using $\alpha = 7$ (see [24, 25]). This is the best gain we can obtain for this scheme using server-aided cryptography.

Strong-secure variant. We next focus on a new variant, denoted $\mathcal{GS}^*_{\mathsf{strong}}$ which is SA-strong-unlinkable. In fact, it seems hard to give a generic strong server-aided secure group signature scheme, based on the generic method defined above. We thus focus on the Delerablée-Pointcheval XS group signature scheme for which it is possible to define such stronger version. In fact, the intermediary should not have access to the CL signature (A, u) and thus should not know the random values from the double El Gamal encryption (for obvious reasons). For the signature of knowledge part, it seems that the intermediary can only perform the computations of the bilinear pairings $e(T_2, g_2)$ and $e(T_2, w)$. The GSIGN procedure of the $\mathcal{GS}^*_{\mathsf{strong}}$ scheme thus consists for the prover to compute the ciphertext (T_1, T_2, T_3, T_4), to send T_2 to $\mathcal{I}$ and, on reception of $e(T_2, g_2)$ and $e(T_2, w)$, to perform U. Therefore, the intermediary may send wrong and well-chosen values such as described in [39] to obtain some information about the secrets of the prover. However, it is possible to overcome this attack if the prover checks whether the received values are of order the one of the group, which costs 2 more modular exponentiations.

Theorem 2. *$\mathcal{GS}^*_{\mathsf{strong}}$ is the best server-aided variant of $\mathcal{GS}$.*

For this new variant, the computational ratio can also be computed. More precisely, if we take again the experiment measures given in Section 2.1, the computation gain for this server-aided variant (taking into account the security check on the received values) is $\frac{13}{11+2\alpha}$

and thus $13/25 = 0.52$ with $\alpha = 7$, using [24, 25]. This is the best gain we can obtain for this scheme using server-aided cryptography.

6 Conclusion

In this paper, we have formalized the model of server-aided cryptography for anonymity, which is suited for group, blind and ring signature schemes. We have presented the best secure adaptations one can achieve on several existing schemes and we have shown that, for some of them, the resulting scheme are not always interesting in practice, in comparison with the original one. This is due to the fact that the server-aided variant needs to be secure. Note finally that in some cases, the time to send data between the user and the intermediary needs to be carefully taken into account[5], which may modify some results.

References

1. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In M. Bellare, editor, *CRYPTO'00*, volume 1880 of *LNCS*, pages 255–270. Springer, 2000.
2. P. Béguin and J.-J. Quisquater. Fast server-aided rsa signatures secure against active attacks. In *CRYPTO '95*, volume 963 of *LNCS*, pages 57–69. Springer, 1995.
3. M. Bellare, H. Shi, and C. Zhang. Foundations of group signatures: The case of dynamic groups. In *CT-RSA 2005*, volume 3376 of *LNCS*, pages 136–153. Springer, 2005.
4. A. Boldyvera, M. Fischlin, A. Palacio, and B. Warinschi. A closer look at pki: Security and efficiency. In *PKC 2007*, volume 4450 of *LNCS*, pages 458–475. Springer, 2007.
5. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In Franklin [29], pages 41–55.
6. D. Boneh, E.-J. Goh, and K. Nissim. Evaluating 2-dnf formulas on ciphertexts. In *TCC*, volume 3378 of *LNCS*, pages 325–341. Springer, 2005.
7. F. Boudot. Efficient proofs that a committed number lies in an interval. In *EUROCRYPT 2000*, pages 431–444, 2000.
8. X. Boyen and B. Waters. Full-domain subgroup hiding and constant-size group signatures. In *PKC 2007*, volume 4450 of *LNCS*, pages 1–15. Springer, 2007.
9. Ernest F. Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In *ACM Conference on Computer and Communications Security 2004*, pages 132–145. ACM, 2004.
10. J. Camenisch, R. Chaabouni, and A. Shelat. Efficient protocols for set membership and range proofs. In *ASIACRYPT 2008*, volume 5350 of *LNCS*, pages 234–252. Springer, 2008.
11. J. Camenisch and A. Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Franklin [29], pages 56–72.
12. Jan Camenisch, Aggelos Kiayias, and Moti Yung. On the portability of generalized schnorr proofs. In *EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 425–442. Springer, 2009.
13. S. Canard, I. Coisel, and J. Traoré. Complex zero-knowledge proofs of knowledge are easy to use. In *ProvSec*, volume 4784 of *LNCS*, pages 122–137. Springer, 2007.
14. S. Canard and M. Girault. Implementing group signature schemes with smart cards. In *CARDIS '02*, pages 1–10. USENIX, 2002.
15. A. Hui Chan, Y. Frankel, and Y. Tsiounis. Easy come - easy go divisible cash. In *EUROCRYPT 1998*, pages 561–575, 1998.
16. N. Chandran, J. Groth, and A. Sahai. Ring signatures of sub-linear size without random oracles. In *ICALP 2007*, volume 4596 of *LNCS*, pages 423–434. Springer, 2007.
17. D. Chaum. Blind signatures for untraceable payments. In D. Chaum, R. L. Rivest, and A. T. Sherman, editors, *CRYPTO 1982*, pages 199–203. Plenum Press, New York, 1982.
18. D. Chaum and T. P. Pedersen. Transferred cash grows in size. In R. A. Rueppel, editor, *EUROCRYPT 1992*, volume 658 of *LNCS*, pages 390–407. Springer, 1992.

[5] For example, the time needed to send values from a SIM card to a mobile phone is quite low in practice.

19. D. Chaum and E. van Heyst. Group signatures. In *EUROCRYPT*, pages 257–265, 1991.
20. David Chaum and Torben P. Pedersen. Wallet databases with observers. In *CRYPTO 92*, volume 740 of *Lecture Notes in Computer Science*, pages 89–105. Springer, 1992.
21. Commission of the European Communities. Commission recommendation on the implementation of privacy and data protection principles in applications supported by radio-frequency identification. http://ec.europa.eu/information_society/policy/rfid/documents/recommendationonrfid2009.pdf, 2009.
22. Ronald Cramer and Torben P. Pedersen. Improved privacy in wallets with observers (extended abstract). In *EUROCRYPT 93*, pages 329–343, 1993.
23. Cécile Delerablée and David Pointcheval. Dynamic fully anonymous short group signatures. In *VIETCRYPT 2006*, volume 4341 of *Lecture Notes in Computer Science*, pages 193–210. Springer, 2006.
24. A. J. Devegili, M. Scott, and R. Dahab. Implementing cryptographic pairings over barreto-naehrig curves. In *Pairing 2007*, volume 4575 of *LNCS*, pages 197–207. Springer, 2007.
25. ECRYPT II. eBACS: ECRYPT benchmarking of cryptographic systems. http://bench.cr.yp.to/index.html, 2008.
26. U. Feige, A. Fiat, and A. Shamir. Zero-knowledge proofs of identity. *J. Cryptology*, 1(2):77–94, 1988.
27. M. Feldhofer, S.Dominikus, and J. Wolkerstorfer. Strong authentication for rfid systems using the aes algorithm. In *CHES 2004*, volume 3156 of *LNCS*, pages 357–370. Springer, 2004.
28. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO'86*, LNCS, pages 186–194. Springer, 1986.
29. M. K. Franklin, editor. *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *LNCS*. Springer, 2004.
30. Eiichiro Fujisaki and Tatsuaki Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *CRYPTO 97*, volume 1294 of *Lecture Notes in Computer Science*, pages 16–30. Springer, 1997.
31. M. Girault and D. Lefranc. Public key authentication with one (online) single addition. In *CHES 2004*, volume 3156 of *LNCS*, pages 413–427. Springer, 2004.
32. M. Girault and D. Lefranc. Server-aided verification: Theory and practice. In *ASIACRYPT 2005*, volume 3788 of *LNCS*, pages 605–623. Springer, 2005.
33. M. Girault, G. Poupard, and J. Stern. On the fly authentication and signature schemes based on groups of unknown order. *J. Cryptology*, 19(4):463–487, 2006.
34. J. Groth. Fully anonymous group signatures without random oracles. In *ASIACRYPT 2007*, pages 164–180, 2007.
35. J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT'08*, volume to appear of *LNCS*. Springer, 2008.
36. E. Hufschmitt and J. Traoré. Fair blind signatures revisited. In *Pairing 2007*, volume 4575 of *LNCS*, pages 268–292. Springer, 2007.
37. A. Kiayias, Y. Tsiounis, and M. Yung. Traceable signatures. In *EUROCRYPT*, volume 3027 of *LNCS*, pages 571–589. Springer, 2004.
38. E. Kiltz. Chosen-ciphertext security from tag-based encryption. In *TCC'06*, volume 3876 of *LNCS*, pages 581–600. Springer, 2006.
39. Chae Hoon Lim and Pil Joong Lee. A key recovery attack on discrete log-based schemes using a prime order subgroupp. In *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 249–263. Springer, 1997.
40. G. Maitland and C. Boyd. Co-operatively formed group signatures. In *CT-RSA 2002*, volume 2271 of *LNCS*, pages 218–235. Springer, 2002.
41. T. Matsumoto, K. Kato, and H. Imai. Speeding up secret computations with insecure auxiliary devices. In *CRYPTO '88*, volume 403 of *LNCS*, pages 497–506. Springer, 1988.
42. T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO'91*, volume 576 of *LNCS*, pages 129–140. Springer, 1991.
43. R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 552–565. Springer, 2001.
44. C.-P. Schnorr. Efficient identification and signatures for smart cards. In G. Brassard, editor, *CRYPTO*, volume 435 of *LNCS*, pages 239–252. Springer, 1989.
45. H. Shacham and B. Waters. Efficient ring signatures without random oracles. In *PKC 2007*, volume 4450 of *LNCS*, pages 166–180. Springer, 2007.
46. M. Stadler, J.-M. Piveteau, and J. Camenisch. Fair blind signatures. In *Eurocrypt '95*, pages 209–219. Springer-Verlag, 1995.

47. Trusted Computing Group. Direct anonymous attestation. http://www.zurich.ibm.com/security/daa/, 2004.
48. E. R. Verheul. Evidence that xtr is more secure than supersingular elliptic curve cryptosystems. *J. Cryptology*, 17(4):277–296, 2004.
49. S. Xu and M. Yung. Accountable ring signatures: A smart card approach. In *CARDIS'04*, pages 271–286. Kluwer, 2004.

A Security Proof for Group Signature Schemes

A.1 Security Agruments on the Generic Group Signature Scheme

We do not give the security proof that our generic group signature scheme is secure in the BSZ model [3] since we mainly focus on the GSign procedure in our description. But one can easily be convinced that it works, assuming that the GJoin protocol is correctly formed[6], by referring to existing practical constructions [23, 1, 5].

A.2 Sketch of Proof of Theorem 1

We have to prove that that $\mathcal{GS}^*_{\mathsf{weak}}$ is SA-weak-secure. The SA-correctness is easily verified and we do not detailed it. The SA-soundness property is verified due to the result given in Section 4 on the used signature of knowledge. In fact, the adversary has no knowledge on the secret x since this is the prover who perform the related computations. Finally, our server-aided scheme is SA-weak-unlinkable since $\mathcal{GS}$ is secure, using the result of Lemma 2.

We next have to prove that if we modify the above protocol to obtain a more interesting computational gain, then the obtained variant is not secure. In fact, the only thing the intermediary may additionally do is to perform the part of the ZKPK which is related to x, which obviously provides a variant without the soundness property. This concludes the proof. □

A.3 Sketch of Proof of Theorem 2

We have to prove that that $\mathcal{GS}^*_{\mathsf{strong}}$ is SA-strong-secure. We assume that the prover checks that the values $e(T_2, g_2)$ and $e(T_2, w)$ are of order p. Moreover, it currently exists no attack, to the best of our knowledge, which permits a fraudulous intermediary to send special values which will permit him to learn some information about the prover, and thus break the soundness (he has all the necessary knowledge) or the unlinkability property (he recognizes some information he has already fraudulously obtained). Thus, under the assumption that the XS group signature is secure [23], our server-aided variant is secure.

Again, a more efficient variant is necessary not SA-strong-unlinkable, for obvious reasons, which concludes the proof. □

B Other Constructions of Group Signature Schemes

In this appendix, we study the server-aided version of some existing group signature in the standard model. We first give some words of the Groth-Sahai work on non-interactive ZKPK [35].

[6] Which is not an easy task, as it can be verified in *e.g.* [23].

B.1 Groth-Sahai Non-Interactive Proofs

A method to obtain non-interactive ZKPK is given by Groth and Sahai [35] in the standard model, in the case of bilinear groups. This work includes the cases we have already dealt with such as public exponents, public group elements, secret exponents and secret group elements but also bilinear group operations (e.g. addition and multiplication of exponents, point addition or scalar multiplication, bilinear map). There are three different instantiations given in [35], depending on the underlying number theoretic assumption: subgroup decision [6], symmetric external Diffie-Hellman [48] or decision linear [5]. Their approach consists in using commitment schemes with homomorphic properties and the role of the prover is to commit on the secrets she knows. Next, the homomorphic properties permit to obtain a commitment on the considered relation. $\mathcal{P}$ finally reveals the randomizer of the commitment, which permits $\mathcal{V}$ to verify the relation. As a consequence, the work of Groth and Sahai only permits proofs of validity and not directly proofs of knowledge. In the server-aided setting, $\mathcal{I}$ can easily produce the commitments on the secrets she knows without being able to learn anything about the other secrets, under the security of the commitment scheme (that is, subgroup decision, symmetric external DH or decision linear).

B.2 Some Other Group Signatures

The group signature scheme described in [34] does not work as the ones we have already studied. More precisely, the group member interacts with the group manager to obtain a certificate on the public key of a certified signature scheme [4] related to a secret key denoted x. The secret key and the certificate are secrets whereas the corresponding public key is known by some authorities. When the prover wants to produce a group signature on a message m, $\mathcal{P}$ first generates a key pair for a one-time signature scheme, signs the message using the generated secret key and certifies the corresponding public key using her user secret key x. The prover next produces a non interactive ZKPK of a certified signature on the verification public key of the one-time signature scheme[7], using the technique of Groth-Sahai . To reach a better level of anonymity on group signature schemes, the prover also encrypts the certificate of the one-time public key using the tag-based encryption scheme of Kiltz [38] and a non-interactive ZKPK that the encrypted value is the same as the one in the previous proof of knowledge.

In the server-aided setting, it is easy to see that the creation of the one-time signature keys and the signature of the message can be done by the intermediary. Moreover, the certification part should be done by the prover. The non interactive ZKPK can be done cooperatively using the technique described above. Finally, the Kiltz encryption together with the non-interactive zero-knowledge proof of knowledge is done entirely by the intermediary since it does not imply secret values which permit to obtain a substantial gain on efficiency. We consequently obtain a weak anonymous server-aided version of the Groth group signature scheme. Again, a strong anonymous version can also be given but with a less interesting gain in terms of efficiency.

[7] The opener, holding the extraction key for the non interactive ZKPK of Groth and Sahai can extract the certificate and thus retrieve the identity of the signer.

C The Case of Blind Signature Schemes

The notion of blind signature scheme has been introduced by Chaum in [17]. Here, an entity obtains a signature on a message of his choice from an authority which has not learn anything about this message. The most important property of this scheme is that after the disclosure of the original message and the corresponding signature, the authority is not able to link this couple with any previous transcript. However, this kind of protocol can be used dishonestly, and that's why Stadler *et al.* have introduced in [46] the notion of fair blind signatures (FBS) where an authority is able to revoke the anonymity of a mandatory or to retrieve the obtained signature from a given protocol between a mandatory and the signer. We now focus on FBS and use the HT model [36] for this purpose.

Fair blind signature procedures. Fair blind signature schemes, with a signer, a user and a judge (to revoke the anonymity) are composed of the following procedures:

- SKeyGen (resp. UKeyGen, JKeyGen) is the key generation algorithm for the signer (resp. user, judge) which outputs $(\mathsf{ssk}, \mathsf{spk})$ (resp. $(\mathsf{usk}, \mathsf{upk})$, $(\mathsf{jsk}, \mathsf{jpk})$);
- BSign is the blind signature protocol bewteen the signer (on input ssk, jpk and upk) and the user (on input usk, jpk, spk and a message m), at the end of which the user obtains a signature σ on m and the verifier store it view trans of the protocol;
- BVerif is the verification of the blind signature σ on a message m, using spk. Let Trans be the set of all signer's transcripts;
- BUserOpen is the operation executed by the judge on input jsk, a message m and a signature σ and which outputs one user public key upk and a proof of guilty τ;
- BSigOpen is the operation executed by the judge on input jsk and a transcript trans and which outputs a message m, a valid signature σ and a proof of guilty τ;
- BJudge is the public algorithm to verify a judge proof τ.

Verification predicate. In the fair blind signature case, we have to defined the server-aided version BSign* of the blind signature protocol. The input in is in this case the message m and the output out is σ. The predicate ϵ_π is satisfied on input m and σ if and only if $\text{BVerif}(\mathsf{spk}, m, \sigma) = 1$.

Correctness predicate. The correctness predicate $\epsilon_\pi^{\mathsf{corr}}$ for a fair blind signature scheme is satisfied if and only if all the following conditions are verified for any message m and any user i:

- $\text{BVerif}(\mathsf{spk}, m, \text{BSig}^*(i(\mathsf{usk}[i], m), \mathcal{I}(\mathsf{spk}, \mathsf{jpk}), \mathcal{V}(\mathsf{spk}, \mathsf{jpk}))) = 1$;
- $\text{BUserOpen}(\mathsf{jsk}, m, \text{GSig}^*(i(\mathsf{usk}[i], m), \mathcal{I}(\mathsf{spk}, \mathsf{jpk}), \mathcal{V}(\mathsf{spk}, \mathsf{jpk}))) = (\mathsf{upk}[i], \tau)$;
- $\text{BSigOpen}(\mathsf{jsk}, \text{GSig}^*(i(\mathsf{usk}[i], m), \mathcal{I}(\mathsf{spk}, \mathsf{jpk}), \mathcal{V}(\mathsf{spk}, \mathsf{jpk})).\mathsf{trans}) = (m, \sigma)$ where m is a message and $\sigma = \text{GSig}^*(i(\mathsf{usk}[i], m), \mathcal{I}(\mathsf{spk}, \mathsf{jpk}), \mathcal{V}(\mathsf{spk}, \mathsf{jpk}))$;
- $\text{BJudge}(\mathsf{jpk}, m, \text{BSig}^*(i(\mathsf{gsk}[i], m), \mathcal{I}(\mathsf{gpk}), \mathcal{V}(\mathsf{gpk})), \mathsf{upk}[i], \tau) = 1$ and
- $\text{BJudge}(\mathsf{jpk}, \text{BSig}^*(i(\mathsf{gsk}[i], m), \mathcal{I}(\mathsf{gpk}), \mathcal{V}(\mathsf{gpk})).\mathsf{trans}, m, \text{BSig}^*(i(\mathsf{gsk}[i], m), \mathcal{I}(\mathsf{gpk}), \mathcal{V}(\mathsf{gpk})), \tau) = 1$.

Soundness predicate. The soundness predicate $\epsilon_{\pi}^{\mathsf{sound}}$ for a fair blind signature scheme is satisfied if and only if, on input m and σ, we have:

- $\text{BVERIF}(\mathsf{spk}, m, \sigma) = 1$ and $\text{BUSEROPEN}(\mathsf{jsk}, m, \sigma) = \perp$ or $\text{BJUDGE}(\mathsf{jpk}, m, \sigma, i, \tau) = 0$ where $(i, \tau) = \text{BUSEROPEN}(\mathsf{jsk}, m, \sigma)$;
- or $\text{BVERIF}(\mathsf{spk}, m, \sigma) = 1$ and for all $\mathsf{trans}_i \in \mathsf{Trans}$, $(m_i, \sigma_i, \tau_i) = \text{BSIGOPEN}(\mathsf{jsk}, \mathsf{trans})$ and also the relation $\text{BJUDGE}(\mathsf{jpk}, \mathsf{trans}_i, m, \sigma, \tau_i) = 0$;
- or on input (m_0, σ_0) and (m_1, σ_1), $\text{BVERIF}(\mathsf{spk}, m_0, \sigma_0) = 1$ and $\text{BVERIF}(\mathsf{spk}, m_1, \sigma_1) = 1$ and it exists $\mathsf{trans} \in \mathsf{Trans}$ such that the relations $\text{BSIGOPEN}(\mathsf{jsk}, \mathsf{trans}) = (m, \sigma, \tau)$, $\text{BJUDGE}(\mathsf{jpk}, \mathsf{trans}, m_0, \sigma_0, \tau) = \text{BJUDGE}(\mathsf{jpk}, \mathsf{trans}, m_1, \sigma_1, \tau) = 1$ hold.

Additional properties. As for group signature scheme, the HT model [36] describes an additional security property for fair blind signature scheme, which is called non-frameability and where $\mathcal{A}$ should not create a proof that an honest user produced a certain valid signature. The introduction of an intermediary in the SA model does not modify this property, which can be find in [36].

Server-aided construction. We now focus on the FBS scheme of Hufschmitt et al. [36] and more precisely in the BSIGN procedure, which works as follows:

- the user first randomly chooses r and s', produces a commitment C on the message m and computes a Paillier encryption E of m, r and s'. She next computes a double El Gamal encryption Δ_1 of $h_1^{s'}$ and sends to the signer (C, E, Δ_1), together with a ZKPK U that all the above values are well-formed;
- the signer next uses the BBS based CL signature scheme on a commitment C' constructed using C, a random s'', and upk so that the obtained signature, denoted S, is a signature on the message $(m, r, \mathsf{upk}, s = s' + s'')$;
- in the second step of the blind signature process, the user makes a double El Gamal encryption Δ_2 on upk, compute h_1^s and produce a signature of knowledge V that (i) she knows a CL signature from the signer on m, r, upk and s, (ii) s is the discrete logarithm of h_1^s in base h_1 and (iii) upk is encrypted in Δ_2 and that she knows the corresponding usk;
- finally, the signature σ is equal to (h_1^s, V, Δ_2)

Note that prior to this protocol, the user must authenticate herself using her secret key and should sign each message before sending it to the verifier.

It is relatively easy to give the best weak-secure server-aided variant of this protocol. More precisely, the main contribution of the prover stands during the second signature of knowledge for the computation of all values relevant to her secret key usk, using the result given in Section 4. However, to ensure the exculpability, the prover must compute on his own the challenge so as to ensure the non modification of the message to sign from a malicious intermediary. Consequently, the prover will only have two modular exponentiations and two modular multiplications to achieve this signature. Furthermore, the two exponentiations can be performed in parallel with the ones computed by the intermediary.

If we take the experiment measures given in Section 2.1, the computation gain for this server-aided variant is $\frac{7}{46+2\alpha}$ and thus $7/60 \approx 0.11$ using $\alpha = 7$. This is the best gain we can obtain for this scheme using server-aided cryptography.

D The Case of Ring Signature Schemes

The notion of *ring signature* schemes has been formalized in [43]. Here an entity can sign a message on behalf of a group that includes himself. Again a verifier is convinced that the signature has been computed by a member of the group but the identity of the signer is not disclosed. A crucial property of a ring signature scheme is that there is no special entity (e.g., no group manager).

Ring signatures are preferred to group signatures in a dynamic setting where a designated authority is not available and in general, where there is no coordination/cooperation between the users. More specifically, a ring signature scheme can be used by only requiring that users own public keys. Instead, a group signature requires a strong set-up assumption that involves the generation of public parameters and procedures for building the group. But the drawback of current ring signature schemes is that their sizes are proportional to the number of group members.

Ring signature procedures. Ring signature schemes are composed of users and a verifier and of the following procedures:

- UKeyGen is the key generation algorithm for the user which outputs $(\mathsf{usk}, \mathsf{upk})$;
- RSign is the ring signature process executed by the signer on input usk, some user public keys $(\mathsf{upk}_1, \cdots, \mathsf{upk}_n)$ and a message m and outputs a signature σ on m within the ring $\{\mathsf{upk}, \mathsf{upk}_1, \cdots, \mathsf{upk}_n\}$;
- RVerif is the verification of the ring signature σ on a message m within the ring $\{\mathsf{upk}, \mathsf{upk}_1, \cdots, \mathsf{upk}_n\}$.

Verification predicate. In the ring signature case, we have to defined the server-aided version RSign* of the ring signature protocol. ϵ_π is satisfied on input $\mathsf{in} = (m, \{\mathsf{upk}, \mathsf{upk}_1, \cdots, \mathsf{upk}_n\})$ and $\mathsf{out} = \sigma$ iff RVerif$(m, \sigma, \{\mathsf{upk}, \mathsf{upk}_1, \cdots, \mathsf{upk}_n\}) = 1$.

Correctness predicate. The correctness predicate $\epsilon_\pi^{\mathsf{corr}}$ for a ring signature scheme is satisfied if and only if RVerif$(m, \text{RSign}(\mathsf{usk}[i], \{\mathsf{upk}_1, \cdots, \mathsf{upk}_n\}, m), \{\mathsf{upk}[i], \mathsf{upk}_1, \cdots, \mathsf{upk}_n\}) = 1$ for any message m and any user i.

Soundness predicate. Let $\mathcal{PK}$ be the set of user's public keys for which the adversary has asked the corresponding secret key (using the Reveal and the Crpt oracles). The soundness predicate $\epsilon_\pi^{\mathsf{sound}}$ for a ring signature scheme is satisfied if and only if, on input m and σ, the relation RVerif$(m, \sigma, \{\mathsf{upk}_1, \cdots, \mathsf{upk}_n\}) = 1$ holds while $\forall j \in [1, n], \mathsf{upk}_j \notin \mathcal{PK}$.

Server-aided construction. In the following, we use the Shacham and Waters ring signature scheme [45] to show an adaptation of such a scheme in the server-aided model. Let us consider a group G of composite order n and $g, h, u', u_1, \cdots, u_k, A, B_0, \hat{A} \in G$ and $\mathcal{H}$ a collision-resistant hash function. Each prover has a secret key sk and a related public key pk (of a variant of the Waters signature scheme). Here is the signature algorithm for a prover $\mathcal{P}_{i^*}$ and a given message m in the standard model:

- $(m_1, \cdots, m_k) = \mathcal{H}(m, \mathsf{pk}_1, \cdots, \mathsf{pk}_l)$ where pk_i denotes the public key of the ring member i (note that each m_i is a bit in $\{0,1\}$).
- Let $f_i = 0$ for all $i \neq i^*$ and $f_{i^*} = 1$, and let $t_i \in_R \mathbb{Z}_n, \forall i$. $\mathcal{P}_{i^*}$ computes for all $i \in [1,l]$, $C_i = (pk_i/B_0)^{f_i} h^{t_i}$ and $\pi_i = ((pk_i/B_0)^{2f_i - 1} h^{t_i})^{t_i}$. Let $t = \sum_{i=1}^{l} t_i$.
- $\mathcal{P}_{i^*}$ chooses $r \in_R \mathbb{Z}_n$ and computes $S_1 = sk.(u' \prod_{j=1}^{k} u_j^{m_j})^r.\hat{A}^t$ and $S_2 = g^r$.
- $\mathcal{P}_{i^*}$ outputs the signature $((S_1, S_2), \{(C_i, \pi_i)_{i=1}^{l}\})$.

In the server-aided model, the most efficient way to proceed (in terms of computation repartition) is when the intermediary computes $\{(C_i, \pi_i)_{i=1}^{l}\}$ and also $\prod_{j=1}^{k} u_j^{m_j}$. In this case, the intermediary has to compute $3l$ modular exponentiations for the $\{(C_i, \pi_i)_{i=1}^{l}\}$ and k more for $\prod_{j=1}^{k} u_j^{m_j}$ and the prover has 3 modular exponentiations to complete the signature. However, using this configuration, we only reach weak security because (i) an adversary impersonating an intermediary is able to break the unforgeability property by obtaining a signature of a message of his choice by replacing the m_j in $\prod_{j=1}^{k} u_j^{m_j}$ and (ii) break the unlinkability one by identifying which member of the ring he is talking with (he must know the f_i to compute $\{(C_i, \pi_i)_{i=1}^{l}\}$).

We consequently prefer another solution where $\mathcal{P}$ also computes $\prod_{j=1}^{k} u_j^{m_j}$, which permits us to prevent the first attack. Furthermore, if we need strong unlinkability, $\mathcal{P}$ can moreover compute two couples (C_i, π_i) where $i = i^*$ is for her own identity and i_0 is another random one. In this case, the intermediary is not able to distinguish which one among the two potential provers (i^* and i_0) has produced the signature. In this case, the computation repartition is 9 modular exponentiations (the exponentiations implying the m_i's are not considered since they belong to $\{0,1\}$) for the prover and $3l - 6$ for the intermediary.

Recently, Chandran, Groth and Sahai [16] have proposed a more efficient ring signature scheme. More precisely, to sign a message, a prover generates one-time signature keys that are used to sign the message and will certify the one-time public key using his Boneh-Boyen signature key. The prover next has to commit to his public key and prove that the committed verification key belongs to the ring. For this purpose, he arranges the ring in a $\nu \times \nu$ matrix where $\nu = \sqrt{l}$ and commits to the row of the matrix. The creation of the one-time signature keys, the signature of the message and the commitment on the rows of matrix can be done by the intermediary whereas the certification, the commitment on the verification key and the proof that this key belongs to the ring are done by the prover. Again, the gain on efficiency is interesting.

Data Synchronization in Privacy-Preserving RFID Authentication Schemes*

Sébastien Canard and Iwen Coisel

Orange Labs, 42 rue des Coutures, 14000 Caen, France

Abstract. Massively deploying RFID systems, while preserving people's privacy and data integrity, is a major security challenge of the coming years. This is why research related to privacy-preserving authentication is growing, including design of schemes, cryptanalysis and security models. In nearly all such schemes secret key cryptography is used, since RFID tags are extremely constrained in time and space, and untraceability is achieved by updating some data at each authentication. Unfortunately, none of them entirely resists to denial of service attacks, those in which the enemy forces updating of the tag and/or the reader by sending fake messages. Moreover, literature lacks a clear and formal way of comparing schemes w.r.t. this kind of attack. In this paper, we introduce a new characterization, called synchronizability. This allows us to, first, establish a relevant model; second, evaluate existing schemes in this model and point out their deficiencies; third, present a new scheme with all desired features.

1 Introduction

Privacy-preserving authentication for RFID tags is a big issue in recent work. There exists a lot of schemes permitting a tag to authenticate itself to a given reader while being anonymous and untraceable for other possible actors. There are several approaches to construct such scheme, depending on the type of cryptographic key is used. Such scheme can for example be based on public key cryptography [3, 6, 14] or using a secret key centralized infrastructure [3].

Another possibility is to use a secret key infrastructure where each tag shares a personal secret key with the reader. The authentication protocol consists then for the tag in proving that it knows a valid secret key while protecting its privacy. One possibility is to use classical cipher-based challenge response protocols but they do not achieve the forward privacy property. Other solutions [2, 5, 9, 11, 16] may have a problem regarding the efficiency since the verification procedure mainly consists in searching the right tag. Obviously, the tag cannot send in clear any information about its identifier. As a consequence, most of existing schemes [2, 5, 9, 11, 16] necessitate testing all the keys in the database until a match is found. These schemes should moreover include randomness to ensure the privacy property, which can be done either by a value sent by the reader [16] or by an update of the key [2, 5, 9, 11].

In this paper, we focus on schemes where the tag shares a personal secret key with the reader and where this key is updated regularly. The studied schemes consequently necessitate synchronization mechanisms to be sure that the same version of the key is used at both sides. And none of the state-of-the-art RFID authentication scheme [11, 2, 5] is at the same time correct (a valid tag is always accepted), sound (a fake tag is always rejected), privacy-preserving (a tag is anonymous, unlinkable and the scheme is forward secure) and resistant to a denial of service attack.

* This work has been financially supported by the European Commission through the IST Program under Contract IST-2002-507932 ECRYPT and by the French Agence Nationale de la Recherche under the RFID-AP project.

Appeared in M. Aigner (Ed.): RFID SEC 2008, pp. 88–99, 2007.

One other remaining problem is that it does not exist any formal mean to compare these schemes regarding first the capacity of an adversary to desynchronize a tag and the reader; second the way the scheme is able to resynchronize a tag and the reader; third the efficiency of the search procedure in the worst case. These comparisons are necessary to compare schemes and to know what are the advantages and the drawbacks of each of them.

In the following, we first describe in Section 2 the general context of our paper and the security model for privacy-preserving RFID authentication schemes. We next contribute to the current work on models for RFID protocols by giving in Section 3 relevant formal definitions and experiments to study and compare privacy-preserving RFID authentication scheme with key update while considering the three above points. We also apply in Section 4 our new method to existing schemes. Finally, we propose in Section 5 a new RFID authentication scheme with approximately the same features as the Dimitriou protocol but with the additional property that our scheme is privacy-preserving.

2 Context and Model

In this section, we set up our model for privacy-preserving authentication schemes.

2.1 Model for Privacy-Preserving RFID Authentication Schemes

A *tag* $\mathcal{T}$ is a transponder identified by a unique identifier ID with limited abilities. A *reader* $\mathcal{R}$, composed of a transreceiver which communicates with tags and a back-end database $\mathcal{D}_\mathcal{R}$, is a more powerful device able to communicate with several tags up to limited distance. $\mathcal{D}_\mathcal{R}$ contains all identifiers ID of valid tags and additional data such as keys[1]. In the following, we call *search procedure* the step that consists to find the right identifier of a tag in the database.

Procedures. A RFID scheme is composed of the following procedures, where s is a security parameter.

- $\mathsf{SetupReader}(1^s)$ is a probabilistic algorithm which generates a key pair (K_S, K_P) for the reader. We assume that s is implicitly specified in K_P.
- $\mathsf{SetupTag}(ID, K_P)$ is a probabilistic algorithm which returns a tag-dependent secret key K_{ID}. (ID, K_{ID}) is added in the reader's database $\mathcal{D}_\mathcal{R}$ containing the whole set of legitimate tags.
- Auth is an interactive protocol π between the reader $\mathcal{R}$ taking as inputs K_S, K_P and $\mathcal{D}_\mathcal{R}$, and a tag $\mathcal{T}$ taking as inputs K_{ID}, ID and K_P. At the end, the reader either accepts the tag and outputs ID or outputs 0.

Definition of the adversary. In all experiments given below, a challenger $\mathcal{C}$ initializes the system and the adversary isgiven the public key K_P of the reader. We distinguish in the following legitimate tags from corrupted one for which the adversary knows the secrets embeded in it. Moreover, the adversary plays any role in the Auth protocol by e.g. deleting or modifying some requests or responses. More precisely, in all below experiments, $\mathcal{A}$ has access at any time (except when stated) to the following oracles.

[1] Note that in some other works, the database is outside the reader.

Appeared in M. Aigner (Ed.): RFID SEC 2008, pp. 88–99, 2007.

– $\mathcal{O}^{\mathsf{CreateTag}}()$**:** adds a tag to $\mathcal{D}_{\mathcal{R}}$ with unique identifier ID and key K_{ID}.
– $\mathcal{O}^{\mathsf{Corrupt}}(ID)$**:** returns K_{ID} and flags this tag as corrupted.
– $\mathcal{O}^{\mathsf{Launch}}()$**:** makes the reader launch the first request of a new Auth protocol instance π.
– $\mathcal{O}^{\mathsf{SendReader}}(m, \pi)$**:** sends a message m to the reader for the protocol π and outputs the response r.
– $\mathcal{O}^{\mathsf{SendTag}}(m, ID)$**:** sends a message m to tag ID and outputs its r.
– $\mathcal{O}^{\mathsf{Return}}(\pi)$**:** outputs the result of the protocol π, that is 0 if the output of the reader during π is 0 and 1 otherwise.
– $\mathcal{O}^{\mathsf{Execute}}(ID)$**:** executes a complete Auth protocol between the reader and the tag ID. Its output is the one of the $\mathcal{O}^{\mathsf{Return}}$ oracle (1 if accepted and 0 is rejected) together with the transcript of the protocol.

2.2 Security Properties

Correctness. This property (also known as the completeness property) says that a legitimate tag is always accepted in the Auth protocol. The formal definition below is derived from [4].

Definition 1 (Correctness). *An RFID system is said to be* correct *if the probability that the reader outputs* 0 *during the* Auth *protocol* π *with a legitimate tag* ID *belonging to* $\mathcal{D}_{\mathcal{R}}$ *is negligible.*

In some cases, it is necessary to define a strong correctness, where the aim of the active adversary is to make rejected a legitimate tag.

– **Strong Correctness Experiment:**
1. At any time of the game, $\mathcal{A}$ chooses a legitimate tag ID.
2. $\mathcal{A}$ launches a request $\mathcal{O}^{\mathsf{Execute}}(ID)$ and obtains as output the bit b.

Definition 2 (Strong Correctness). *An RFID system is said to be* strong correct *if the probability that* $b = 0$ *at the end of the Strong Correctness Experiment is negligible.*

Soundness. This property states that a fake tag cannot be accepted by the system. It corresponds to the strong soundness in [4] where the adversary can corrupt tags.

– **Soundness Experiment:**
1. At any time of the game, $\mathcal{A}$ plays the role of a tag in the Auth protocol by using successful calls to the $\mathcal{O}^{\mathsf{SendReader}}$ oracle, and a final call to the $\mathcal{O}^{\mathsf{Return}}$ oracle.
2. The experiment's output is 1 if $\mathcal{A}$ is accepted during the Auth protocol and the outputted tag is not corrupted, and 0 otherwise.

Definition 3 (Soundness). *An RFID system is* sound *if the probability that the bit* b *returned by the* $\mathcal{O}^{\mathsf{Return}}$ *oracle at the end of the Soundness Experiment is equal to 1 is negligible.*

Appeared in M. Aigner (Ed.): RFID SEC 2008, pp. 88–99, 2007.

Privacy. A tag should be anonymous and untraceable for everyone except the valid reader. Moreover, the scheme has to preserve the privacy of a tag in its previous authentications, even if an adversary compromises it and outputs its internal data: this is what is called forward-privacy. Many formal definitions concerning privacy in RFID systems have been proposed so far [1, 7, 8, 15, 12, 13] and we use here a definition which is closed to the ones in [7, 13].

All these features are described in the following experiment, where the goal of the adversary $\mathcal{A}$ is to recognize one tag among two.

– **Privacy Experiment:**
1. At any time of the game, $\mathcal{A}$ chooses two tags ID_0 and ID_1 in the set of legitimate tags and sends (ID_0, ID_1) to $\mathcal{C}$.
2. $\mathcal{C}$ randomly chooses a bit b. The tag ID_b is called the *challenge* tag. ID_0 and ID_1 are withdrawn from $\mathcal{D}_{\mathcal{R}}$ and thus cannot be manipulated by the adversary using oracles. ID_b is added to $\mathcal{D}_{\mathcal{R}}$, as an exact copy of the tag ID_0 or ID_1.
3. Again, $\mathcal{A}$ interacts with the whole system through all oracles. Note that $\mathcal{A}$ can interact with the challenge tag without any restriction.
4. $\mathcal{A}$ finally outputs a bit b'.

Definition 4 (Privacy). *We say that an RFID scheme has the* privacy *property if the probability that $b = b'$ differs from $1/2$ by a fraction that is at most negligible.*

3 New Characterizations for RFID Schemes

3.1 On RFID Schemes with Synchronization

We now consider RFID authentication schemes where the reader shares a secret key with each tag and where this key is updated after each (not necessarily successful) authentication. In these schemes, it may be possible to desynchronize several times a tag and the reader by e.g. forcing the tag to update its key whereas the reader does not, inducing a more important work by the reader during the search procedure.

In the RFID world, it exists in the literature several ways to fight against this possible desynchronization. One may design a scheme with a good resynchronization mechanism such that no matter being the number of times the tag and the reader are desynchronized, the system will always resynchronize. But this method can be subject to denial of service attack (see [11] and Section 4.1). A way to prevent this drawback is to limit the number of accepted resynchronization by the reader and reject a tag after that. But this may imply that a valid tag is rejected if it has been desynchronized a sufficient number of time (see [2] and Section 4.1). Another possibility for a scheme is to prevent a tag and the reader to be desynchronized more than a fixed number of times, so that the number of resynchronization is also limited (see [5] and Section 4.2).

As a consequence, since a scheme can use one of the above technique, it seems difficult to compare related work. In this paper, we construct a model to quantify the quality of a RFID authentication scheme when considering first the number of desynchronization the scheme is subject to; second the number of resynchronization the scheme can manage; third the efficiency of the search procedure in case of multiple desynchronizations.

Appeared in M. Aigner (Ed.): RFID SEC 2008, pp. 88–99, 2007.

3.2 Notation and Definition

Let us consider a valid RFID tag ID with the corresponding key denoted K_{ID}. We need to introduce new notation that will be used in the rest of the paper.

First, to take into account the update of the key K_{ID} during the authentication protocol, we will denote by $K_{ID}^{(0)}$ the key output by the $\mathsf{SetupTag}(ID)$ procedure, by $K_{ID}^{(j)}$ the j-th version of the key of the tag ID after updating it j times.

Second, as the key is stored both in the reader and the tag, as described in the $\mathsf{SetupTag}$ procedure, we need to distinguish the key that is embedded into the tag, denoted TK_{ID}, and the key stored in the database, denoted RK_{ID}, both initialized to $K_{ID}^{(0)}$. Usually $TK_{ID} = RK_{ID} = K_{ID}^{(*)}$ but, as the tag and the reader can be desynchronized by some adversaries, we may have $TK_{ID} = K_{ID}^{(i)}$ and $RK_{ID} = K_{ID}^{(j)}$ with $i \neq j$. We can now define a desynchronization as following.

Definition 5. *Considering an arbitrary tag ID with $TK_{ID} = K_{ID}^{(i)}$ and reader with $RK_{ID} = K_{ID}^{(j)}$, a* desynchronization *is an operation inducing an incrementation of the value $|i-j|$ by 1. The value $|i-j| \in \mathbb{N}$ is called* number of desynchronizations.

3.3 The Desynchronization Capacity

We first study in depth the *desynchronization* characterization: we want to know for a given scheme the maximum number of desynchronizations between a tag and the reader an adversary can create. We want to compute the corresponding scale, called *the desynchronization value*.

More precisely, to have a better characterization, we will distinguish on one side the maximum number of desynchronization $D_{\mathcal{R}}$ an adversary can create when only focusing on the reader, and on the other side this maximum number $D_{\mathcal{T}}$ when $\mathcal{A}$ only focuses on the tag. The desynchronization value consequently corresponds to the couple $(D_{\mathcal{R}}, D_{\mathcal{T}})$.

In fact, the Strong Correctness Experiment presented in Section 2.2 gives us a direct way to define more formally and compute these values. Let $\mathcal{A}$ being the adversary playing the experiment. At the end of step 1, $\mathcal{A}$ chooses a tag ID. We denote by $TK_{ID} = K_{ID}^{(i)}$ (resp. $RK_{ID} = K_{ID}^{(j)}$) the tag's version (resp. reader's version) of K_{ID} at the end of this step. The maximum number of desynchronization obtained by the adversary $\mathcal{A}$, when $\mathcal{A}$ only focuses on the tag (resp. the reader), is $D_{\mathcal{T},\mathcal{A}} = j - i$ (resp. $D_{\mathcal{R},\mathcal{A}} = i - j$).

The desynchronization value can now be defined more formally as follows.

Definition 6. *For a given RFID authentication scheme, the* desynchronization value *of a scheme is the couple $(D_{\mathcal{R}}, D_{\mathcal{T}})$ with $D_{\mathcal{R}} = \underset{\mathcal{A}}{Sup}(D_{\mathcal{R},\mathcal{A}})$ and $D_{\mathcal{T}} = \underset{\mathcal{A}}{Sup}(D_{\mathcal{T},\mathcal{A}})$. The scheme is said* $(D_{\mathcal{R}}, D_{\mathcal{T}})$–desynchronizable.

Appeared in M. Aigner (Ed.): RFID SEC 2008, pp. 88–99, 2007.

3.4 The Resynchronization Capacity

Now, we study the *resynchronization* characterization: we study the capacity a scheme has to resynchronize a tag and the reader by computing the corresponding scale, called *the resynchronization value*, defined as follows.

Definition 7. *For a given RFID authentication scheme, when considering an arbitrary valid tag ID with $TK_{ID} = K_{ID}^{(i)}$ and a reader with $RK_{ID} = K_{ID}^{(j)}$, we denote by $R_{\mathcal{R}}$ (resp. $R_{\mathcal{T}}$) the maximum number of desynchronizations the scheme can tolerate to accept the tag ID during the* Auth *procedure, if only the tag (resp. the reader) has been updated, i.e. after the* Auth *procedure $RK_{ID} = K_{ID}^{(j)}$ (resp. $TK_{ID} = K_{ID}^{(i)}$). The* resynchronization value *of the scheme is the couple $(R_{\mathcal{R}}, R_{\mathcal{T}})$ and the scheme is said* $(R_{\mathcal{R}}, R_{\mathcal{T}})$–resynchronizable.

We now give methods to compute respectively $R_{\mathcal{R}}$ and $R_{\mathcal{T}}$. Let us consider a tag $\mathcal{T}$, the reader $\mathcal{R}$ (synchronized with $\mathcal{T}$) and a counter C which will be incremented in each round of the two experiments. We first introduce two procedures: $\mathsf{UpdateTag}(ID)$, which forces TK_{ID} to be updated, and $\mathsf{UpdateReader}(ID)$ which forces RK_{ID} to be updated.

The computation of $R_{\mathcal{R}}$ (resp. $R_{\mathcal{T}}$) works as follows. During round C, we produce C desynchronizations of the tag (resp. the reader) using $\mathsf{UpdateTag}(ID)$ (resp. $\mathsf{UpdateReader}(ID)$) and then launch the Auth procedure between $\mathcal{R}$ and $\mathcal{T}$. If the reader accepts the tag, TK_{ID} and RK_{ID} are resynchronized (i.e. $TK_{ID} = RK_{ID}$), C is incremented and a new round is started[2]. Else we stop the algorithm and output the value $C - 1$ which is exactly $R_{\mathcal{R}}$ (resp. $R_{\mathcal{T}}$).

3.5 Conclusions on the Synchronization

Given a RFID authentication scheme, we are now able to compare both desynchronization and resynchronization scales. Intuitively, if the desynchronization value is less or equal to the resynchronization value, an adversary will not be able to win the Strong Correctness Experiment by desynchronizing a tag or a reader: the scheme is considered secure.

Definition 8. *For a given RFID authentication scheme, if $D_{\mathcal{R}} \leq R_{\mathcal{R}}$ and $D_{\mathcal{T}} \leq R_{\mathcal{T}}$, the scheme is said* synchronizable. *Else, the scheme is said* desynchronizable.

Note that if an adversary $\mathcal{A}$ is able to desynchronize a tag (or a reader) more times than the scheme is able to resynchronize it, the scheme is obviously not strong correct.

3.6 Efficiency of the Search Procedure

During the protocol, the reader performs a search procedure in order to retrieve the identifier of the tag it is interacting with. The efficiency of a scheme is strongly connected to the one of this procedure. If some cases, an attacker can saturate a reader with bad requests inducing a denial of service attack.

[2] the tag (resp. the reader) will be updated once more.

Consequently, we need to compute, for a given RFID authentication scheme, the number of operations that are performed by the reader in the worst case to either retrieve an identifier or reject a tag. In the following, an operation corresponds to one of the following items[3].

- The call to a no-key mathematical or cryptographic function (e.g. a hash function, a modular reduction, a pseudo-random function).
- The call to a symmetric key cryptographic function (e.g. a HMAC).

4 Study of Existing Schemes

In the rest of the paper, we assume that all hash functions are cryptographically secure. More precisely, we need them to be one-way (i.e. given $H(m)$, it is infeasible to retrieve m) and collision-free (i.e. given $H(m)$, it is computationally infeasible to find $m' \neq m$ s.t. $H(m) = H(m')$) and we also use the random oracle. $H^k(m)$ denotes the k-composite of H (i.e. $H \circ \ldots \circ H$ k times).

In the following, we do not necessary detail the proofs of our theorems. This is due to the fact that most of these results are already known in the cryptographic community and also due to lack of space. Our aim in this section is to give the result of our analysis for existing schemes.

We here study existing and known secure schemes. For example, our study does not include the proposal of Lim and Kwon [10] and the one of Le et al. [8] since they have been recently shown as traceable [13], even if the second one [8] has interesting features concerning desynchronization and resynchronization.

4.1 The OSK Schemes

The OSK scheme has been introduced by Ohkubo, Suzuki and Kinoshita in 2003 [11]. Let H_1, H_2 be two secure hash functions. Each tag ID of the system is set up with a secret key $TK_{ID} = K_{ID}^{(0)}$ and the database is set up with all the couples $(ID, RK_{ID} = K_{ID}^{(0)})$. In the life cycle of the tag ID, the key $K_{ID}^{(j)}$ is updated as $K_{ID}^{(j+1)} = H_2(K_{ID}^{(j)})$ after each (successful) authentication. During the protocol, the tag ID simply response $\mathcal{H}_1(TK_{ID})$ to a request from the reader and then update its key.

The search procedure of the OSK protocol is very simple and works as follows. For each entry $i \in \mathcal{D}_{\mathcal{R}}$, the reader computes $H_1(RK_i)$ and compares it with r. If there is no match, the reader temporarily updates the entries of $\mathcal{D}_{\mathcal{R}}$ and starts again the search procedure. After a match is found, the key of the corresponding tag is updated.

Theorem 1. *The OSK scheme is* $(\infty, 0)$*–desynchronizable,* $(\infty, 0)$*–resynchronizable and the search procedure works in an infinite number of operations in the worst case.*

[3] The search procedure also includes equality tests but, as they are negligible compared to the other kinds of operation, we neglect them.

Appeared in M. Aigner (Ed.): RFID SEC 2008, pp. 88–99, 2007.

The OSK$_m$ Scheme. This protocol aims at increasing the efficiency of the search procedure by stopping it after after m updates, where m is fixed.

Theorem 2. *The OSK$_m$ scheme is $(\infty, 0)$–desynchronizable, $(m, 0)$–resynchronizable and the search procedure works in $2m + 1$ operations per tag in the worst case. The scheme is consequently desynchronizable.*

The OSK-AO Scheme. Avoine and Oechslin have proposed in [2] another modification of the OSK protocol os as to reduce the complexity of the search procedure at the cost of a larger database. Let n be the size of the database and m be a security parameter. Due to space restriction, we refer the reader to [2] for details.

Theorem 3. *The OSK-AO scheme $(\infty, 0)$–desynchronizable, $(m-1, 0)$–resynchronizable and the search procedure works in $2(t-1)^2/n$ operations per tag in the worst case. The scheme is consequently desynchronizable.*

4.2 The Dimitriou Scheme

The Dimitriou protocol has been proposed in [5] and is presented in Figure 1. As the tag only updates its secret key if it has authenticate the reader, this scheme has not the drawback of the OSK like schemes, at the cost of more rounds. Note that contrary to [5], we need that the reader stores ID, the current and the last versions of the key, in order to prevent an attack against the strong correctness property.

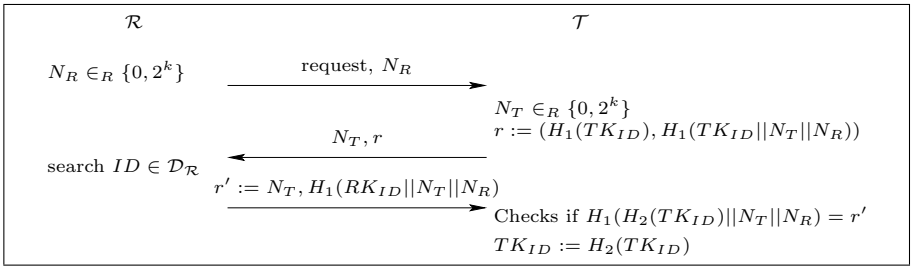

Fig. 1. Protocol of Dimitriou

In the following, we denote $r_1 = H_1(TK_{ID})$ and $r_2 = H_1(TK_{ID}||N_T||N_R)$. During our new search procedure, the reader compares the hash of each RK_{ID} with r_1. If a match is found and if r_2 is well constructed, it outputs ID and updates RK_{ID} and RK'_{ID}. If no match is found, the reader compares the received with all previous keys RK'_{ID} and does the same as before, except updating the keys.

Theorem 4. *The Dimitriou scheme is $(0, 1)$–desynchronizable, $(0, 1)$–resynchronizable and the search procedure works in 2 operations per tag in the worst case. The scheme is consequently synchronizable.*

Appeared in M. Aigner (Ed.): RFID SEC 2008, pp. 88–99, 2007.

Proof.

- **Desynchronization:** tn adversary cannot update the tag since he cannot produce $H_1(RK_{ID}||N_T||N_R)$: $D_{\mathcal{R}} = 0$. Moreover, if the adversary block the last message of a valid protocol, the reader updates the key whereas the tag does not. As this can be done only once by the adversary, $D_{\mathcal{T}} = 1$: the scheme is $(0, 1)$–desynchronizable.
- **Resynchronization:** if the tag is updated, there is now way for the reader to make the link between the received value and the values stored in $\mathcal{D}_{\mathcal{R}}$. The tag is thus rejected and $R_{\mathcal{R}} = 0$. As described above, if the database is updated only once, the tag is accepted. On the other hand, if RK_{ID} is updated twice, TK_{ID} is no longer stored and ID is rejected: $R_{\mathcal{T}} = 1$, the scheme is $(0, 1)$–resynchronizable and so synchronizable.
- **Search procedure efficiency:** in the worst case, the search procedure computes the hash of all keys in the actual and in the previous version: there are 2 operations per tag.

Note that the efficiency of the search procedure can be improved by directly storing the two versions of the key so that the reader has no operation to perform (only tests). The drawback is that the size of the memory is multiplied by two.

The main problem of the Dimitriou, whatever be the version, is that it is traceable w.r.t. the privacy experiment described in Section 2 since the adversary can call the $\mathcal{O}^{\mathsf{SendTag}}((request, N_R), ID_0)$ oracle for tag ID_0 (to obtain the value $H_1(TK_{ID_0})$) and choosing at random ID_1. If the response from the challenge contains $H_1(TK_{ID_0})$, $\mathcal{A}$ outputs $b' = 0$ and $b' = 1$ otherwise.

5 Our Proposal: the C^2 Scheme

5.1 Description of the C^2 Scheme

Our scheme, presented on Figure 2, aims at proposing a secure construction and being good regarding synchronisation and efficiency as defined in Section 3. As in the Dimitriou scheme,

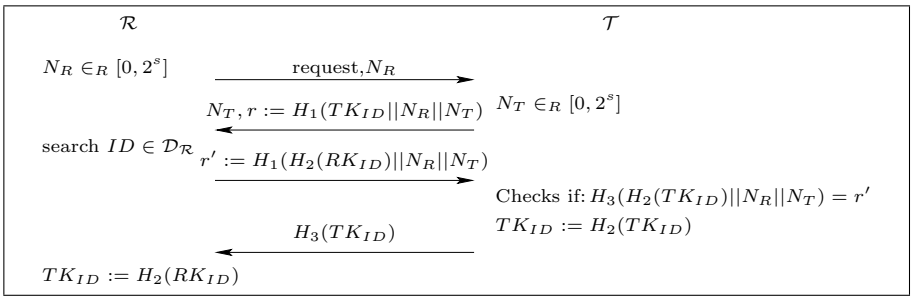

Fig. 2. The C^2 Protocol

the random values N_R and N_T are used to protect the scheme against replay attack. We next protect the scheme on privacy at the cost of a less efficient search procedure. We next design our scheme in such a way that the tag updates its key only after having authenticated the reader. Finally, the last message from the tag permits to convince the reader that the tag has updated its internal key.

In the search procedure, for each $ID \in \mathcal{D}_{\mathcal{R}}$, the reader computes the value $H_1(RK_{ID}||N_R||N_T)$ and compares it with r. In case of a match, the procedure outputs the corresponding identifier. Else, for each $ID \in \mathcal{D}_{\mathcal{R}}$, the reader computes the ephemeral identifier $EID = H_2(RK_{ID})$, computes $H_1(EID||N_R||N_T)$ and compares it with r. In case of a match, the reader outputs the corresponding identifier and updates RK_{ID} (so $RK_{ID} = TK_{ID}$). Otherwise, the tag is rejected.

We thus obtain a $(1, 0)$-synchronizable scheme instead of $(0, 1)$-synchronizable for the Dimitriou scheme. It seems better in practice as we do not need to store the previous identifier of the tag and so use less memory for the database.

5.2 Security Arguments

We first prove that the C^2 scheme is secure.

Theorem 5. *The C^2 scheme is sound under the security of the hash function and in the random oracle.*

Proof. To win the Soundness Experiment, an adversary can first guess which random value N_R will be sent. If $\mathcal{A}$ has beforehand asked the tag the corresponding answer: this is not possible if the security parameter s is well chosen. Another possibility for the adversary is to produce a valid message $H_1(TK_{ID}||N_R||N_T)$ without knowing a valid (and uncorrupted) value TK_{ID}: this is not possible under the security of the hash function. So the scheme is sound.

Theorem 6. *The C^2 scheme is private under the security of the hash function and in the random oracle.*

Proof. The scheme clearly provides the anonymity of the tag under the assumption that the hash function is one-way. Moreover, the scheme is unlinkable since $\mathcal{A}$ has no control on the value N_T. Again, if the used hash functions are secure, the three first messages are useless for $\mathcal{A}$. One possibility for $\mathcal{A}$ is to use $H_3(TK_{ID})$ but this message is only sent when the key is updated: $\mathcal{A}$ cannot learn any information of it. Finally, the scheme provide the forward-privacy property using the same argument. So the scheme is private.

Theorem 7. *The C^2 scheme is $(1, 0)$–desynchronizable, $(1, 0)$–resynchronizable and the search procedure works in 3 operations per tag in the worst case. The scheme is consequently synchronizable.*

Proof.

– **Desynchronization:** Here we first highlight the fact that $\mathcal{A}$ is not able to produce valid messages for this protocol. Indeed, the only way to do this is to know the secret key used either by the tag or the reader. As the tag is uncorrupted and the hash function is one-way, $\mathcal{A}$ cannot learn anything about this key. By blocking the last message of a protocol, $\mathcal{A}$ desynchronize the tag as it updates its secret key contrary to the reader. $\mathcal{A}$ cannot use this technique twice as the reader resynchronize its key on reception of the

Appeared in M. Aigner (Ed.): RFID SEC 2008, pp. 88–99, 2007.

second message (during the search procedure). As a consequence, $\mathcal{A}$ must produce the third messages without interacting with the reader. As said before it is impossible and so $D_{\mathcal{R}} = 1$. The only way for $\mathcal{A}$ to force the update of RK_{ID} without updating TK_{ID} is to produce the last message. As $\mathcal{A}$ has no information about TK_{ID}, he is not able to produce such a message. So, $D_{\mathcal{T}} = 0$. Finally, the scheme is $(1, 0)$–desynchronizable.

- **Resynchronization:** By definition of the scheme, the tag is still accepted if TK_{ID} is updated once. This is not the case if it is updated twice. So, $R_{\mathcal{R}} = 1$. If the reader updates the stored keys once, TK_{ID} is no longer stored in the database and the reader does not find a match, even if it updates all the stored keys once. As a consequence the tag is rejected, $R_{\mathcal{T}} = 0$. The scheme is $(1, 0)$–resynchronizable and so synchronizable.
- **Search procedure efficiency:** On reception of a random value, the search procedure computes, for all $ID \in \mathcal{D}_{\mathcal{R}}$, $H_1(RK_{ID}||N_R||N_T)$, with $RK_{ID} = K_{ID}^{(*)}$ and its update $K_{ID}^{(*+1)}$. Next, it compares these values with the received one. This implies per tag 3 calls to a hash function (two for the computation of messages and one for the update).

Note that the search procedure can be fastened by storing in the database the updated key and obtain 2 operations in the search procedure.

6 Conclusion

Inspired by the state of the art, we have suggested a new model for RFID authentication most adapted for schemes using a secret key infrastructure with an update key mechanism and we have designed a protocol with good properties while being secure in the privacy model.

Acknowledgments. We are grateful to Marc Girault for his suggestions of improvement, and to anonymous referees for their valuable comments.

References

1. G. Avoine. Adversarial model for radio frequency identification. Cryptology ePrint Archive, Report 2005/049, 2005.
2. G. Avoine and P. Oechslin. A scalable and provably secure hash based RFID protocol. In *PerSec 2005*. IEEE Computer Society Press, 2005.
3. B. Calmels, S. Canard, M. Girault, and H. Sibert. Low-cost cryptography for privacy in rfid systems. In *CARDIS 2006*, volume 3928 of *LNCS*, pages 237–251. Springer, 2006.
4. I. Damgård and M. Ø. Pedersen. Rfid security: Tradeoffs between security and efficiency. In *CT-RSA 2008*, volume 4964 of *LNCS*, pages 318–332. Springer, 2008.
5. T. Dimitriou. A lightweight RFID protocol to protect against traceability and cloning attacks. In *SecureComm 2005*. IEEE Computer Society Press, 2005.
6. M. Girault and D. Lefranc. Public key authentication with one (online) single addition. In *CHES 2004*, volume 3156 of *LNCS*, pages 413–427. Springer, 2004.
7. A. Juels and S. A. Weis. Defining strong privacy for rfid. In *PERCOMW '07*, pages 342–347, Washington, DC, USA, 2007. IEEE Computer Society.
8. T. V. Le, M. Burmester, and B. de Medeiros. Universally composable and forward-secure rfid authentication and authenticated key exchange. In *ASIACCS 2007*, pages 242–252. ACM, 2007.
9. S. Lee, T. Asano, and K. Kim. RFID mutual authentication scheme based on synchronized secret information. In *Symposium on Cryptography and Information Security*, Hiroshima, Japan, January 2006.
10. Chae Hoon Lim and Taekyoung Kwon. Strong and robust rfid authentication enabling perfect ownership transfer. In *ICICS*, volume 4307 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2006.

Appeared in M. Aigner (Ed.): RFID SEC 2008, pp. 88–99, 2007.

11. M. Ohkubo, K. Suzuki, and S. Kinoshita. Cryptographic approach to "privacy-friendly" tags. In *RFID Privacy Workshop 2003*, 2003.
12. K. Ouafi and R. C.-W. Phan. Traceable privacy of recent provably-secure rfid protocols. In *ACNS 2008*, volume 5037 of *LNCS*, pages 479–489, 2008.
13. R.-I. Paise and S. Vaudenay. Mutual authentication in rfid: security and privacy. In *ASIACCS 2008*, pages 292–299. ACM, 2008.
14. S. Vaudenay. RFID privacy based on public-key cryptography (abstract). In *ICISC 2006*, volume 4296 of *LNCS*, pages 1–6, Busan, Korea, November-December 2006. Springer-Verlag.
15. S. Vaudenay. On privacy models for rfid. In *ASIACRYPT 2007*, pages 68–87, 2007.
16. S. Weis, S. Sarma, R. Rivest, and D. Engels. Security and privacy aspects of low-cost radio frequency identification systems. In *SPC 2003*, volume 2802 of *LNCS*, pages 454–469. Springer-Verlag, 2003.

Appeared in M. Aigner (Ed.): RFID SEC 2008, pp. 88–99, 2007.