Université de Montréal

Calculs Multipartites

par

Anton Stiglic

Département d'informatique et de recherche opérationnelle

Faculté des arts et des sciences

Mémoire présenté à la Faculté des études supérieures
en vue de l'obtention du grade de

Maître ès sciences (M.Sc.)

juin, 2000

Université de Montréal

Faculté des études supérieures

Ce mémoire intitulé

Calculs multipartites

présenté par :

Anton Stiglic

a été évalué par un jury composé des personnes suivantes :

(président-rapporteur)

Geňa Hahn

(directeur de recherche)

Claude Crépeau

(membre du jury)

Pierre McKenzie

Mémoire accepté le :

Sommaire

Ce mémoire traite des calculs multipartites, un artifice cryptographique initialement suggéré par A. C. Yao dans [Yao82]. Dans son article il nous propose le problème du millionnaire qui se résume comme suit : deux millionnaires, disons Alice et Bob, veulent savoir lequel d'entre eux est le plus riche. Comme tout bon riche, ni Alice, ni Bob, ne veulent dévoiler à personne le montant précis de leur fortune. Yao donna une solution permettant à Alice et à Bob de satisfaire leur curiosité en respectant ces contraintes. Ce résultat a donné lieu à une généralisation appelée protocole de calcul multipartite (PCM). Dans un PCM, nous avons un nombre quelconque de participants p_1, p_2, ..., p_N, ayant chacun une donnée privée, respectivement, d_1, d_2, ..., d_N. Les participants veulent calculer la valeur d'une fonction publique $f(x_1, x_2, ..., x_N)$ au point $(d_1, d_2, \ldots, d_N)$. Un PCM sera dit *sûr* si aucun participant ne peut rien apprendre de plus que ce qu'il peut déduire de sa propre entrée et de la valeur du résultat du calcul global, sous des conditions particulières dépendant du modèle utilisé. Comme la plupart des protocoles cryptographiques, la sécurité d'un PCM est soit calculatoire (c'est à dire qu'elle dépend d'une hypothèse cryptographique) ou inconditionnelle (c'est-à-dire qu'elle est sécuritaire contre un adversaire ayant une puissance de calcul illimitée). Dans ce mémoire, nous traiterons surtout des protocoles qui sont inconditionnellement sûrs. Nous élaborerons sur ce qui a été fait dans le passé pour en venir à présenter un protocole qui est un hybride entre deux nouveaux protocoles : le premier provenant d'Adam Smith et de l'auteur de ce mémoire ([SS98]), l'autre provenant de

Cramer, Damgård, Dziembowski, Hirt et Rabin ([CDD+99]). Ce dernier protocole est le PCM inconditionnellement sûr le plus efficace, en complexité d'exécution, connu à ce jour.

Mots Clés : Calculs Multipartites, Cryptologie, Partage de Secret, Sécurité Inconditionnelle, *Span Programs*.

Table des matières

Table des protocoles

Table des figures

À ma belle fiancée et amour éternel, Marcela Quiroz.

Remerciements

J'aimerais tout d'abord remercier Adam Smith, avec qui j'ai eu de nombreuses discussions enrichissantes sur le sujet de ce mémoire. Son talent et sa douance à comprendre et à résoudre des problèmes continuera toujours à m'épater. J'aimerais également remercier mon directeur de recherche, Claude Crépeau, qui m'a initié à la fascinante science cryptologique.

Je tiens aussi à souligner mon contentement de l'apport académique du département d'informatique et de recherche opérationnelle de l'Université de Montréal, et plus particulièrement des enseignants avec qui j'ai eu le privilège de collaborer ou discuter tout au long de ma formation universitaire. Je pense entre autres à Claude Crépeau, Rudolf Keller, Michael Florian, Neil Stewart, Jean Vaucher, Jian-Yun Nie, Pierre McKenzie et Paul Bratley. Leurs dévouements à la recherche, leurs enthousiasmes et leurs qualités didactiques m'ont été d'une grande inspiration.

Finalement, j'aimerais exprimer ma reconnaissance à ceux qui m'ont donné des commentaires, suggestions et corrections bien appréciés lors de la rédaction de ce mémoire : Paul Dumais (le revendicateur philosophe), Frédéric Légaré (le savant perdu), Alain Tapp (le savant fou), Christian Paquin (le *hacker-cracker*), Sébastien Paquet (le physicien par excellence), Marcela Quiroz (mon mentor grammaticale bien-aimée) et Jean-François Raymond (le *super raquette*).

Chapitre 1

Introduction

Récemment, l'Internet a subit une énorme expansion. Cette entité est passée du domaine académique à une tendance qui est aujourd'hui acceptée et utilisée par la population en général. Pour réaliser l'ampleur de cette croissance, il suffit de mettre en lumière quelques statistiques, comme par exemple, le nombre d'ordinateurs hôtes sur le réseau international qui double à chaque jour, ou le nombre de pages Web qui augmente au rythme d'un million par jour !

Avec cette fulgurante révolution on assiste à une prospérité des bienfaits : l'accès à des milliers de ressources de façon instantanée, la publication à une échelle de grandeur autrefois inimaginable, l'abolition des frontières de communication en général et encore plus. Par contre, la subordination de tous les services par l'Internet et la dépendance de plus en plus marquée des individus envers ces services amènent souvent, aussi, la perte de l'intimité personnelle des usagers ainsi que la possibilité de dénégation des services à ces derniers comme nous allons voir plus loin. Voilà deux problématiques de l'Internet que la science cryptologique tente de résoudre. Comment assurer la confidentialité des données privées des individus et comment assurer un service continu, fiable et surtout honnête ? Un individu se branchant sur l'Internet pour la première fois ignore très probablement que ses habitudes de navigation peuvent être enregistrées, qu'un site Web peut conserver des données à propos d'un individu dans l'ordinateur même de l'individu (en utilisant ce qu'on appele des *cookies*) afin d'accumuler des statistiques sur celui-ci. Un usager novice de l'Internet n'est probablement pas au courant non plus du fait que son adresse de courrier électronique peut être connue par un site Web visité, que tous les messages qu'un individu envoit à un *news group* sont en fait archivés pendant plusieurs années et peuvent être lus par n'importe quel autre individu.

L'information personnelle vaut très cher pour les "business de l'Internet" et les grandes entreprises sont prêtes à entreprendre des démarches extrêmes pour cueillir cettre information. Nous pensons par exemple à l'affaire récente des numéros de série des processeurs d'ordinateur *Pentium III (PIII)* (un processeur est un engin central au fonctionnement d'un ordinateur). La compagnie *Intel* a incorporé un numéro de série unique dans chaque processeur *PIII*, qui fait partie de la majorité des ordinateurs vendus de nos jours. Ce numéro de série est enregistré au moment de la vente, donc sont associés à ceci des informations personnelles de l'acheteur (comme par exemple le nom, prénom, addresse, numéro de téléphone, peut-être même le numéro de carte de crédit, l'âge, le sexe, l'occupation et encore plus). Le numéro de série peut ensuite être reconnu par un site Web sur lequel l'acheteur navigue. Soudainement nous avons la possibilité d'associer les habitudes navigationnelles non seulement à un individu quelconque, mais à une personne méticuleusement identifiée. Cela n'est qu'un exemple d'une notion plus générale connue sous le nom de *Big Brother*. Un *Big Brother* est une entité qui surveille et enregistre le comportement de tout le monde, c'est une entité qui brise la confidentialité des données privées des individus. Cette notion est même apparue sur les écrans de cinémas d'Hollywood, nous pensons par exemple au film *The Net*, dans lequel une femme se fait changer son identité au point de ne plus pouvoir fonctionner dans la vie quotidienne.

La cryptologie nous apporte des solutions : la confidentialité des données, l'anonimité et la pseudonimité. Un individu peut garder privé de yeux épieux l'information qu'il juge personnelle en utilisant ses schèmes d'encryption, il peut aussi camoufler ses données tout en l'utilisant pour des fonctionnalités publiques comme dans le cas des calculs multipartites où un individu veut garder une certaine valeur privée tout en l'incluant dans un calcul publique. L'anonimité, par contre, permet à un individu d'utiliser les fonctionnalités de l'Internet sans se faire identifier. L'anonimité est en fait un droit constitutionnel dans plusieurs pays, dont le Canada et les États-Unis. Nous pouvons très bien combiner l'anonimité avec le camouflage des données. La pseudonimité, elle, permet à un individu de fonctionner sous un pseudonyme, sans que ce dernier soit lié à la vraie identité de l'individu. La pseudonymité possède l'avantage de permettre à quelqu'un de se bâtir des références et d'établir une certaine confiance de la part de ses interlocuteurs. La pseudonymité est fait en vigueur dans la vie courante, lorsque nous allons acheter un carton de lait à l'épicerie du coin par exemple, le caissier peut nous reconnaître par le biais de nos caractéristiques faciales, notre physique, notre voix, sans pour autant connaître notre nom véritable, notre occupation, notre numéro d'assurance sociale ou d'autres informations personnelles. La pseudonimité est aussi habituelle dans le monde artistique et littéraire. Nous citons par exemple le poète qui publiait sous le pseudonyme de Pablo Neruda sans dévoiler sa vraie identité, ou par exemple les textes Fédéralistes, publiés sous le pseudonyme de Publius. La pseudonimité a même fait le sujet des récits telle que l'excellente oeuvre futuristique de John Brunner, *The Shockwave Rider*, publiée en 1975, qui semblait déjà saisir les possibles problèmes d'un monde subordonné par un réseau informatique et où le héro est un personnage pouvant incarner différentes identités (différents pseudonymes).

L'autre point de vulnérabilité consiste en une dépendance à un service fourni par un réseau. Cette dépendance sous-entend que l'on doit posséder une certaine confiance en l'authenticité de ce qu'on utilise et que l'on doit croire en sa fiabilité. Comment peut-on savoir qu'une certaine fonctionnalité fait vraiment ce qu'elle prétend faire, comment peut-on savoir que l'on peut utiliser cette fonctionnalité de façon consistante ? Cette vulnérabilité provient du fait que les services d'un réseau sont susceptibles à des attaques malicieuses, commises par des entités, connues sous le nom générique d'adversaire. Plusieurs types d'adversaire co-existent et les calculs multipartites, par exemple, se caractérisent par les types d'adversaires contre lesquels ils sont sécuritaires. Nous pouvons reconnaître la sévérité réelle d'une attaque en se rappelant du virus de Morris en novembre 1988 qui a bouleversé l'Internet, qui à cette époque était consistué d'approximativement 60000 ordinateurs. Le virus avait paralisé une majeure partie de l'Internet, de Cambridge, Massachusetts, Berkeley, California, New-Jersey, jusqu'aux laboratoires de recherche de la NASA, de Los Alamos et d'autres établissement de recherche, des Universités et des bases militaires. Quoiqu'aucune donnée n'a été détruite suite à l'attaque, le coût total du rétablissement des systèmes a été estimé dans l'ordre d'un million à 100 millions de dollars. Cet événement a prouvé la vulnérabilité de l'Internet. Des schèmes cryptographiques peuvent aider à contrer de telles malignités. Le développement des méthodes de calculs multipartites, en particulier, aident à l'avancement de ces techniques.

1.1 Cryptologie et les calculs multipartites

Le domaine de la cryptologie fut marqué par une grande évolution au cours des deux dernières décennies. En effet, de nombreux nouveaux protocoles de chiffrement, d'authentification, de signature digitale et autres ont vu le jour. Ces procédés cryptographiques ont tous eu comme premier modèle le scénario où une paire de participants, Alice et Bob, veut exercer une certaine fonctionnalité en présence d'adversaires (corrupteurs, imposteurs, espions ou autres). Mais avec la maturité des techniques cryptographiques et la croissance des applications due à l'avènement du calcul distribué, le besoin fondamental d'avoir un modèle où nous pouvons distribuer à un plus grand nombre de participants les "capacités cryptographiques" ([DDFY94]) est maintenant reconnu. Les protocoles de calculs multipartites (PCM) sont constitués d'un tel modèle de participants distribués. Ils sont d'ailleurs la solution à des problèmes tels que le vote distribué (e.g. [CFSY96, FOO93]), les encans à distance (e.g. [FR96]), le partage de protocoles de chiffrement et de signature (e.g. [DDFY94, GJKR96a, GJKR96b]), les requêtes de données privées (e.g. [CGKS95, KO97, IK99, Ray00]), etc... En gros, un PCM permet à un groupe de N participants de calculer $Y = F(x_1, \ldots x_N)$, où F est une fonction connue de tous et $x_1, \ldots, x_N$ sont les entrées privées des participants. Un PCM sera dit *sûr* si aucun participant ne peut apprendre plus que ce qu'il pourrait déduire de sa propre entrée, de la définition de F et de la valeur Y (une définition beaucoup plus formelle à été donnée dans [Can95]). Dans un PCM nous modélisons un adversaire comme une entité centralisée contrôlant un certain nombre de participants. La façon dont ce contrôle s'établit, la manière de le définir et le système (réseau) de communication utilisé par les participants sont des caractéristiques permettant de distinguer les PCM.

Définitions, notations, modèles

Nous définissons quelques termes et entités utilisés dans les descriptions de ces modèles.

- P désigne l'ensemble des participants.
- $N = |P|$ est le nombre de participants.
- S est l'ensemble de secrets.
- D est l'initiateur (de l'anglais : *Dealer*) d'un protocole.
- k est le paramètre de sécurité pour un protocole.
- $x \in_R X$ signifie que x est un élément choisi de façon uniformément aléatoire de l'ensemble X.
- K est un corps fini, où $|K| \geq 2^k$.

- $w \in K$ est une racine primitive dans K.
- **Participant :** Une machine de Turing ayant accès à une source de bits aléatoires.
- **Synchronisme :** Une horloge globale. On pose que n'importe quel message envoyé à un "tic" de l'horloge atteint sa destination au "tic" suivant de l'horloge.
- **Adversaire passif**, $t < \theta$

  Un adversaire centralisé qui peut seulement lire la mémoire de $t < \theta$ participants.
- **Adversaire actif**, $t < \theta$

  Un adversaire qui peut lire la mémoire et corrompre les protocoles (*trojan horse*) des adversaires qu'il contrôle (*Byzantine errors*).
- **Adversaire échec**, $t < \theta$

  Un adversaire qui, pour les participants qu'il contrôle, peut interrompre la communication de ceux-ci avec le reste des joueurs (*Fail-Corrupt Adversary*). Ce type d'adversaire ne peut pas lire les mémoires des participants qu'il contrôle.
- **Adversaire statique :** Un adversaire qui choisit les participants qu'il va contrôler *avant* le début des calculs.
- **Adversaire dynamique :** Un adversaire qui peut choisir les participants *au cours* du processus de calcul.

Dans le cas où nous avons un PCM inconditionnellement sûr, l'adversaire a une puissance de calcul infinie.

Nous disons qu'un protocole est (θ, N)-*privé* si un adversaire passif $t < \theta$ ne peut apprendre de l'information à propos des valeurs d'entrées des participants honnêtes. Par ailleurs, nous dirons qu'il est $(\theta, N) - robuste$ (*résilient*) si un adversaire actif $t < \theta$ ne peut apprendre de l'information à propos des valeurs d'entrées des participants honnêtes et il ne peut engendrer un résultat du calcul multipartite qui soit cohérent du point de vue individuel de chaque participant honnête tout en étant incohérent avec l'ensemble des valeurs des entrées des participants honnêtes.

Nous étudierons surtout les PCM opérant avec des canaux de communication sûrs, l'hypothèse la plus caractéristique du modèle à sécurité inconditionnelle. Dans ce contexte, chaque paire de participants possède un canal de communication sûr (i.e. l'adversaire ne peut lire, modifier ou générer aucun message sur ces canaux). D'autres modèles de communication mettent en jeu les canaux à sécurités conditionnelles (avec protocoles de chiffrement et d'authentification), les canaux dits non-sécuritaires (n'utilisant que des protocoles d'authentification, les messages étant envoyés en clair) et les canaux non-authentifiés où l'adversaire possède le contrôle total de la communication (peut effacer, générer et modifier des messages à volonté). L'intérêt des canaux de communication sûrs est de permettre une vue plus abstraite du problème de PCM et d'arriver à une solution à sécurité inconditionnellement sûre. La plupart des protocoles que nous étudierons auront aussi accès à un canal de diffusion publique, ceci procure un avantage que nous verrons plus loin. Dans nos discussions, le réseau de communication sera toujours synchronisé. Dans un réseau asynchrone, un temps arbitraire (mais fini) peut s'écouler avant l'arrivée d'un message. Ce scénario apporte une plus grande complexité. Une excellente étude de PCM avec réseau asynchrone est la thèse de Canetti [Can95].

Nous ne discuterons pas non plus des adversaires dynamiques qui apportent aussi une plus grande complexité. Plusieurs protocoles ont été présumés sûrs contre des adversaires dynamiques mais il a été prouvé par la suite que tel n'était pas le cas (voir par exemple [BH92], [Can95], [CFGN96], [CDD$^+$99]).

Finalement, notons que tout PCM est composé d'un sous-protocole appelé partage de secret vérifiable (PSV) (parfois le protocole PSV est implicite, comme dans [CCD88]). Un protocole PSV permet de mettre en partage un secret parmi plusieurs participants de façon à ce que les participants puissent vérifier que le secret est bien mis en partage sans divulguer leurs parts (les valeurs servant à reconstituer le secret partagé). Nous donnerons une définition formelle des PSV au chapitre 2.

1.2 Travaux antérieurs

Yao ([Yao82]) a été le premier à suggérer le problème des PCM. Il a donné un PCM pour deux joueurs satisfaisant certains critères de confidentialité et de robustesse (pas identiques à nos définitions) avec probabilité d'erreur. La sécurité de ce PCM est basée sur la sécurité des fonctions de chiffrement à clé publique.

Goldreich, Micali et Wigderson ([GMW87]) ont donné une première solution générale à sécurité conditionnelle basée sur l'hypothèse de l'existence de permutation à trappe. Ils ont présenté un protocole $(N, N)-$*privé*, ainsi qu'un protocole $(N/2, N) - $ *robuste* dans un modèle à réseau synchrone avec canaux de communication non-sécuritaires.

Par la suite, Chaum, Damgård et van de Graaf ([CDvdG87]) ont présenté une solution hybride contre un adversaire actif qui permet à un des participants de se sécuriser de façon inconditionnelle. Ce fut le premier pas vers la découverte d'un PCM à sécurité inconditionnelle.

Plus tard, Chaum, Crépeau et Damgård ([CCD88]) ont trouvé, de façon indépendante à Ben-Or, Goldwasser et Wigderson ([BGW88]), une solution à sécurité inconditionnelle dans le scénario de canaux de communications sûrs. Ces premiers auteurs ont donné un protocole $(N/3, N)-$*robuste* ainsi qu'un protocole $(N/2, N)-$*privé*. Cependant, les protocoles utilisent des preuves à divulgation nulle, qui par leur nature même induisent une probabilité d'erreur (pouvant, par contre, être exponentiellement petite par rapport au paramètre de sécurité choisi). Les auteurs de [BGW88] donnèrent aussi, dans un même scénario, un protocole $(N/2, N)-$*privé*, avec probabilité d'erreur exponentiellement petite, mais leur protocole $(N/3, N)-$*robuste* n'avait aucune probabilité d'erreur. Ils ont aussi démontré qu'il était impossible d'avoir un protocole parfait (sans probabilité d'erreur) lorsque l'adversaire est actif et contrôle au moins $N/3$ participants, ou $N/2$ des participants dans le cas d'un adversaire passif.

Avec l'addition d'un canal de diffusion sûr au modèle précédent, Rabin et Ben-Or ([RB89]) et de façon indépendante Beaver [Bea90], ont présenté une solution $(N/2, N) - robuste$ inconditionnellement sûre avec probabilité d'erreur exponentiellement petite. Cela est le mieux que l'on puisse faire contre un adversaire à seuil d'après les résultats de [BGW88] et puisque qu'il est même impossible de mettre en partage un secret de façon vérifiable (un PSV) si $N/2$ ou plus des participants sont corrompus.

Chaum [Cha90] a présenté une solution hybride qui pour certains aspects repose sur des hypothèses *calculatoires*, mais qui pour d'autres est inconditionnellement sûre. Le protocole est sûr contre un adversaire actif corrompant a participants ou un adversaire passif corrompant p joueurs (l'adversaire doit choisir d'être soit actif, soit passif), pour autant que $2a + p < n$.

Fitzi, Hirt et Maurer [FHM98] (et la version corrigée [FHM99]) ont introduit le type d'adversaire échec. Dans la même ligne de pensée que Chaum, ils ont présenté un PCM robuste contre un adversaire qui corrompt passivement p joueurs, activement a joueurs et contrôle e participants échec. Toute combinaison de possession, en même temps, est permise pour autant que $2a + 2p + e < n$, ceci dans un réseau synchrone avec canal de diffusion publique. Si $3a + 2p + e < n$, il existe un protocole *sans aucune* probabilité d'erreur. Dans le cas d'un réseau sans canal de diffusion publique un protocole existe, avec probabilité d'erreur négligeable, pour autant que $3a + e < n$.

Hirt et Maurer [HM97] ont caractérisé un type d'adversaire plus général que celui à seuil et ont présenté des protocoles privés et robustes contre ceux-ci. L'adversaire général se caractérise par un ensemble constitué des ensembles de joueurs que l'adversaire peut contrôler. Smith et Stiglic [SS98], parallèlement à Cramer, Damgård, Dziembowski, Hirt et Rabin [CDD+99], ont présenté des solutions dans le même contexte que [HM97], mais plus efficaces, le PCM de [CDD+99] étant optimal. Les deux résultats s'inspirent grandement des travaux de [RB89].

1.3 Organisation des chapitres

Ce mémoire se veut une synthèse des PCM inconditionnellement sûrs. Notre but consiste à présenter un protocole hybride entre celui de [SS98] et [CDD+99], qui sont l'état de l'art des PCM inconditionnellement sûrs. Quoique moins efficace que celui de [CDD+99], le protocole que nous présenterons est beaucoup plus intuitif et unique en son genre. Pour arriver au résultat, nous décrirons, en ordre chronologique, les principaux sous-protocoles et protocoles qui ont contribué aux idées des PCM de [SS98] et [CDD+99]. Nous présenterons aussi des preuves et un formalisme général qui est souvent manquant dans la littérature de ce domaine.

Le chapitre suivant traite des protocoles de partage de secret vérifiable (PSV). Une définition formelle en est donnée, et les notions de sécurité qui s'y rapportent sont abordées. Nous y présentons le protocole classique de partage de secret (PS) de Shamir [Sha79] ainsi que le PSV de Feldman [FM88] et celui de [CDD+99] (qui est une modification de celui de Feldman).

Au chapitre 3, nous présentons un PCM construit à l'aide du PSV de [CDD+99] et de quelques protocoles qui ont été introduits dans [BGW88], [SS98] ainsi que d'autres que nous construirons. Le PCM obtenu, quoique pas le plus efficace qui existe jusqu'à aujourd'hui, est original, simple et utile pour des raisons pédagogiques. Nous analyserons tous les sous-protocoles utilisés.

Par la suite, nous introduirons au chapitre 4 les structures d'adversaires généralisés. Pour ce faire, nous définirons la notion de *span programs*, un modèle de calcul servant à généraliser les PS. Nous décrirons un protocole de PS (tiré de [KW93]), utilisant les *span programs*, qui permet de généraliser le PCM présenté.

Finalement, dans la conclusion, nous résumerons les contributions techniques qu'apporte ce mémoire et nous discuterons à propos des problèmes encore ouverts.

Chapitre 2

Partage de secret (PS) et partage de secret vérifiable (PSV)

Un protocole de partage de secret PS$-(t, N)$ permet à un *initiateur* de mettre en partage parmi N participants un secret s de façon à ce qu'un ensemble de t participants puissent calculer le secret s, mais qu'un ensemble de participants de moindre cardinalité n'ait aucune information (dans le sens de Shannon [Sha48]) à propos du secret. Par contre, dans un tel protocole un initiateur malfaisant pourrait mettre en partage un secret de façon non cohérente, c'est-à-dire de façon à ce qu'une fois la phase de distribution terminée, un groupe de participants A (avec $|A| \geq t$) qui se regrouperait pour retrouver le secret calculerait une valeur s_0, tandis qu'un autre groupe $B \neq A$ (avec $|B| \geq t$) calculerait une autre valeur $s_1 \neq s_0$. Lorsqu'une telle situation peut se présenter, il faut pouvoir permettre aux participants de *vérifier* que le secret est bien distribué de façon cohérente. Il peut aussi arriver que certains participants tentent de corrompre la phase du calcul du secret en donnant des parts qui ne proviennent pas réellement de l'initiateur, il nous faut donc une technique pour contrer cela aussi. Le protocole de partage de secret vérifiable (PSV), introduit par Chor, Goldwasser, Micali et Awerbuch dans [CGMA85], est un artifice cryptographique qui répond à ces deux problèmes. Nous présentons dans ce chapitre les protocoles PS (plus particulièrement celui de Shamir) ainsi que le PSV de Feldman ([BGW88] et [FM88]) et une modification de celui-ci, retrouvé dans [CDD+99], qui permet à un plus grand nombre de joueurs malhonnêtes de participer. Nous présentons aussi des protocoles et des notions que ces derniers utilisent, notamment les codes correcteurs et les protocoles de vérification d'information.

2.1 Partage de secret (PS) de Shamir

La majorité des PSV se basent sur la notion de partage de secret (PS) de Shamir [Sha79] (un protocole de PS a été trouvé indépendamment par [Bla79], mais ce dernier est moins utile dans notre contexte). Un protocole de partage de secret est constitué d'une paire de protocole $(Dist, Rec)$, où $Dist$ permet à un initiateur D de distribuer un secret s aux participants de P et Rec est le protocole de reconstitution du secret, exécuté par les participants sans la présence de D.

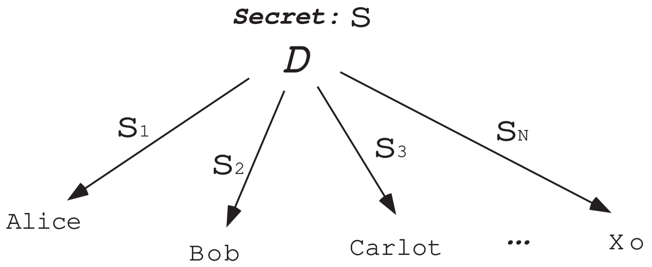

FIG. 2.1: Partage de secret

Un PS-(t, N) est un protocole PS qui permet à un ensemble de t participants de reconstruire le secret sans donner aucune information à un groupe de moins de t participants (c'est un protocole (t, N)-*privé*).

Le protocole PS-(t, N) de Shamir se décrit comme suit :

Protocole[PS de Shamir]

Distr(D, P, s) :

- D choisit un polynôme aléatoire $p \in_R K[x]$ de degré au plus $t-1$ tel que $p(0) = s$ et dont les coefficients sont choisis de façon uniformément aléatoire dans K, où $|K| > N$ (la taille de K peut même être exponentiellement grande en N sans que le protocole ne devienne inefficace).
- D choisit N éléments distincts, non nuls, $a_1, \ldots, a_N \in K^+$ et les dévoile publiquement. Il distribue ensuite $\alpha_i := p(a_i)$ au joueur p_i, pour $1 \leq i \leq N$.

Rec(P) :

- Chaque joueur diffuse publiquement sa valeur α_i.
- Prendre t valeurs diffusées et interpoler un polynôme f de degré au plus $t-1$.
- Retourner la valeur $s = f(0)$.

♠ L'intuition derrière le fonctionnement de ce protocole est de camoufler le secret dans le terme constant d'un polynôme aléatoire de degré $t-1$ et de distribuer N points distincts de ce polynôme. Un tel polynôme ne pouvant être interpolé qu'avec t points ou plus, le secret sera sécuritairement dissimulé contre une coalition de moins de t participants.

Théorème 2.1.1 *Dans le protocole de PS de Shamir, n'importe quel groupe de plus que t participants peut, facilement, reconstruire le secret de façon unique, mais aucun groupe de $t-1$ participants ou moins ne peut en tirer de l'information.*

Preuve: Supposons que nous avons t parts, $\alpha_{i_1}, \ldots, \alpha_{i_t}$.
Nous cherchons s tel que :
$f(x) = s + c_1x + \ldots + c_{t-1}x^{t-1}$ et $f(a_{i_1}) = \alpha_{i_1}, \ldots, f(a_{i_t}) = \alpha_{i_t}$.

Ceci revient à résoudre le système d'équations linéaires :

$$\underbrace{\begin{pmatrix} 1 & a_{i_1} & {a_{i_1}}^2 & \dots & {a_{i_1}}^{t-1} \\ & & \vdots & & \\ 1 & a_{i_t} & {a_{i_t}}^t & \dots & {a_{i_t}}^{t-1} \end{pmatrix}}_{A} \begin{pmatrix} s \\ c_1 \\ \vdots \\ c_{t-1} \end{pmatrix} = \begin{pmatrix} \alpha_{i_1} \\ \alpha_{i_2} \\ \vdots \\ \alpha_{i_t} \end{pmatrix}$$

Dans ce système à t équations et t inconnues, $s, c_1, \dots, c_{t-1}$, nous reconnaissons A sous la forme d'une matrice de VanDermonde, dont le déterminant bien connu est :

$$\det A = \Pi_{1 \leq j < k \leq t}(a_{i_k} - a_{i_j})$$

Les a_{i_j} sont distincts, par définition du protocole, le déterminant est donc non nul et la solution est unique ! Notons que l'ajout d'autres équations n'affecte en rien le calcul de s. La reconstitution du secret se fait donc efficacement par un simple calcul linéaire.

Si par contre, nous avions $u < t$ équations, nous obtiendrions un système où les c_i peuvent s'écrire en fonction de s et chaque $s \in K$ engendre une solution possible et le nombre de solution pour chaque s est le même. Si $c_{i_1}, \dots, c_{i_{t-1}}$ étaient choisis de façon uniformément aléatoire, nous n'aurions aucune information à propos de s (et de façon plus générale, si nous avions de l'information à propos de s à priori, nous n'obtiendrions aucune information supplémentaire à propos de s). $\square$

Il est à noter que nous pouvons interpoler la fonction de façon plus directe par la formule d'interpolation de Lagrange :

$$p(x) = \sum_{j=1}^{t} \alpha_{i_j} \Pi_{1 \leq k \leq t, k \neq j} \frac{x - a_{i_k}}{a_{i_j} - a_{i_k}}$$

Nous vérifions cette relation en remplaçant x par a_{i_l}, ce qui annule tous les termes de la somme sauf pour $j = l$ qui vaut α_l. Nous avons donc un polynôme de degré au plus $t - 1$ qui passe par les points (a_{i_j}, α_{i_j}), $1 \leq j \leq t$. En sachant que le polynôme est unique, nous avons ainsi une expression pour ce polynôme.

2.2 Partage de secret vérifiable (PSV)

Un protocole PSV de sa part est constitué d'une paire de protocoles $(Distr, Rec)$, où

- $Dist$ est un protocole de partage de secret (souvent une extension du protocole de partage de secret (PS) de Shamir). Chaque joueur p_i calcule ver_i à la fin de ce protocole ($ver_i = 1$ si p_i croit que le secret a été distribué de façon cohérente, $ver_i = 0$ autrement).
- Rec est un protocole de reconstitution du secret.

Ces protocoles doivent vérifier les critères suivants :

- **Cohérence**
  - Si un joueur honnête, p_i, calcule $ver_i = 1$ à la fin de $Dist$, alors $ver_j = 1$ pour tous les autres joueurs honnêtes p_j.
  - Si l'initiateur D est honnête, $ver_i = 1$ pour tout joueur honnête p_i.
- **Confidentialité**
  - Si un secret s est choisi de façon uniformément aléatoire dans S et D est honnête, un adversaire A n'apprendra aucune information à propos de s si aucun joueur honnête n'a encore commencé Rec.
- **Reconstitution**
  - Si un participant honnête p_i calcule $ver_i = 1$ à la fin de $Dist$, alors il existe une valeur $\sigma \in S$, fixée à la fin de $Dist$, telle que tous les participants honnêtes calculent σ à la fin de Rec sauf avec probabilité d'erreur $< 2^{-k}$. De plus, si D est honnête alors $\sigma = s$.

Lorsque nous disons qu'un certain joueur J n'a aucune information à propos d'un élément s, nous voulons dire, plus formellement, que

$$\forall_{f:\sum^* \to \{0,1\}} \forall_n \; Pr[f(Vue(J)_n) = 1] = Pr[f(V_n) = 1]$$

où $Vue(J)_n$ est la distribution des vues (voir définition plus loin) possibles de J, (après les diverses exécutions possibles du protocole sous les diverses entrées possibles) et V_n est la distribution des vues d'un participant n'ayant aucune information *à priori* relativement à s. En d'autres mots, la vue de P peut être simulée par un simulateur n'ayant aucune information à propos de s. Nous cherchons une sécurité inconditionnelle !

La vue d'un participant P consiste en l'ensemble des données qui ont été enregistrées (publiquement ou en privé) et qui ont pu être observées par P. Plus formellement, dans un modèle de Machine de Turing, la vue d'un joueur P est caractérisée par les rubans contenant sa source aléatoire, son ruban d'entrée, de sortie et d'écriture ainsi que des rubans, partagés, d'entrées communes et de communication privée avec les autres joueurs. Le modèle de Machine de Turing d'un calcul bipartite, ainsi que les rubans composant le modèle sont illustrés par la figure 2.2.

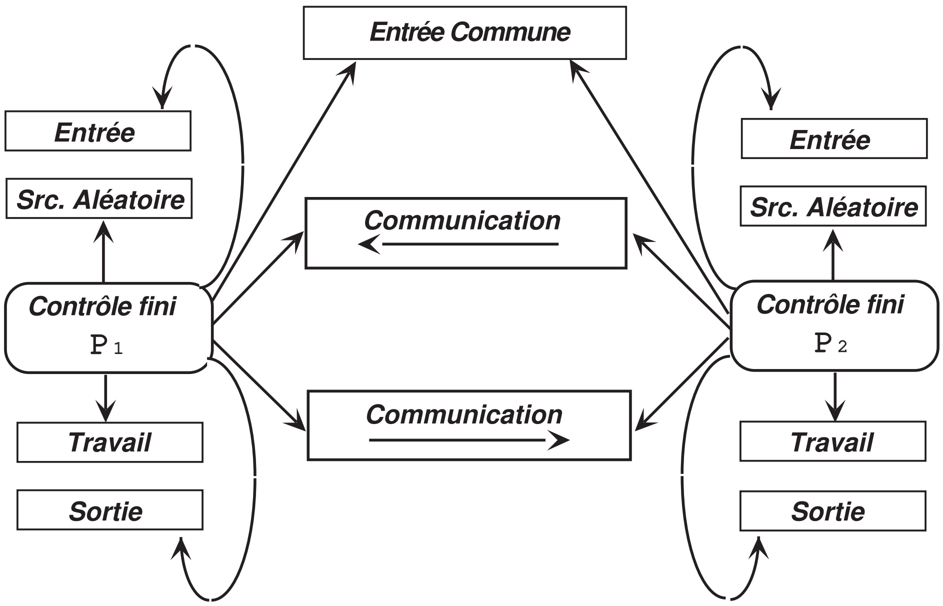

FIG. 2.2: Modèle de Turing

Nous n'élaborerons plus sur les détails de ce modèle puisqu'ils ne sont pas utilisés dans le reste de ce mémoire.

Notons pour ce qui suit, que dans un PSV un initiateur est aussi considéré comme un participant (contrairement au protocole de PS). Remarquons de plus que pour un PSV $(N/k, N) - robuste$ nous poserons $N = kt + 1$ et metterons en partage le secret à l'aide d'un polynôme de degré au plus t (pas $t - 1$ comme pour les protocoles de PS). De cette façon, nous pouvons considérer que le nombre de participants malhonnêtes est au plus t (ce qui est moindre que N/k) et que ceux-ci ne peuvent reconstituer le secret par eux-mêmes.

2.3 Partage de secret vérifiable (PSV) de Feldman

Le premier PSV que nous présentons a été introduit pour la première fois dans [BGW88], c'est un protocole qui est dû en grande partie à Paul Feldman. Le protocole tel que présenté initialement est $(N/3, N) - robuste$, une modification de celui-ci (que nous présenterons après) s'avère être $(N/2, N) - robuste$.

Pour comprendre le protocole, nous devons d'abord présenter les codes correcteurs qui y sont utilisés.

2.3.1 Codes correcteurs et décodeur Berlekamp-Welch

Sans trop aller dans les détails des codes correcteurs nous allons décrire les concepts de base de ceux-ci pour pouvoir ensuite présenter un théorème sur la correction des polynômes.

L'objectif des codes correcteurs est d'ajouter de la redondance aux messages que nous devons transmettre à travers un canal bruyant (causant des erreurs) de façon à pouvoir rectifier le message reçu à sa valeur initiale. Dans le cadre des protocoles PSV, le canal bruyant est tout simplement un objet métaphorique, les erreurs possibles proviennent en fait des participants malhonnêtes. Un code linéaire C, sur K, est un sous-espace vectoriel de K^n. On dénote $[n, k, d]$ un code C de dimension k sur K^n où d est le minimum de l'ensemble des distances de *Hamming* entre deux mots de code distincts (la distance de Hamming entre deux mots m_1, m_2 est le nombre de coordonnées où m_1 et m_2 diffèrent). Un code ayant une distance de Hamming d peut corriger jusqu'à $\lfloor (d-1)/2 \rfloor$ erreurs.

Théorème 2.3.1 (Decodeur Berlekamp-Welch) *Étant donnés m pairs de points $(x_i, s_i) \in K \times K$ (x_i distincts) tel qu'il existe un polynôme P de degré au plus t tel que pour toutes sauf k valeurs de i on a $s_i = P(x_i)$, où $2k + t < m$, nous pouvons retrouver P de façon efficace.*

Preuve: voir [GS92] □

L'algorithme proposé dans [GS92] donne lieu à une preuve d'exactitude très simple mais n'est pas le protocole optimale. Un protocole plus efficace, $O(m)$, est présenté par Berlekamp and Welch [BW].

Donc, avec $N = 3t + 1$, si nous partageons un secret en utilisant un polynôme de degré au plus t avec N $= 3t + 1$ participants, nous pouvons tolérer jusqu'à t participants malhonnêtes (les participants honnêtes peuvent *corriger* les mauvaises valeurs lors de la reconstitution du secret).

2.3.2 Partage de secret vérifiable de Feldman

Nous utiliserons N points distincts de K, notés $1, 2, \ldots, N$, pour la mise en partage des secrets. Ces points peuvent être obtenus par un élément primitif (ω) de K, via l'exponentation ($1 := \omega^0, 2 := \omega^1, \ldots, N := \omega^{N-1}$ par exemple). Par cette dernière façon de faire, la multiplication est directe ($w^i \cdot w^j = w^{(i+j) \bmod |K|-1}$) et l'addition peut se faire à l'aide du polynôme primitif associé à ω (en travaillant modulo un certain polynôme irréductible). Peu importe le choix, nous gardons de

façon abstraite la description des opérations d'addition et de multiplication dans le corps, comme auparavant.
Ceci dit, nous pouvons maintenant définir le protocole de PSV de Feldman.

Protocole[PSV de Feldman]

Distr(D, P, s) :

1. D choisit un polynôme *bivarié* (i.e. à deux variables) $f(x, y)$ aléatoire de degré au plus t en chaque variable, tel que $f(0, 0) = s$. Nous construisons la matrice

$$\begin{pmatrix} s_{11} & s_{12} & \dots & s_{1i} & \dots & s_{1N} \\ & & & \vdots & & \\ & & & \vdots & & \\ s_{i1} & s_{i2} & \dots & s_{ii} & \dots & s_{iN} \\ & & & \vdots & & \\ s_{N1} & s_{N2} & \dots & s_{Ni} & \dots & s_{NN} \end{pmatrix}$$

   où $s_{ij} = f(i, j)$. La rangée i définit $f_i(y)$ et la colonne i définit $g_i(x)$. Les $g_i(0)$ (qui ne sont pas représentés dans la matrice) sont les coefficients qui peuvent être interpolés pour retrouver s comme dans le protocole de PS de Shamir, les autres valeurs ne servent qu'à des fins de vérification de cohérence entre les participants.

2. D donne $a_{i1} = s_{i1}, \dots, a_{iN} = s_{iN}$ et $b_{1i} = s_{1i}, \dots, b_{Ni} = s_{Ni}$ au joueur p_i, pour $i = 1, \dots, N$.

3. Le joueur p_i vérifie que les deux ensembles $a_{i1}, \dots, a_{iN}$ et $b_{1i}, \dots, b_{Ni}$ obtenus de D, interpolent bien en des polynômes $f_i(x), g_i(x)$, de degrés au plus t. Si cela n'est pas le cas, p_i vote pour la disqualification de D.

4. Chaque p_i donne a_{ij} au joueur p_j.
   Noter que si l'initiateur est honnête, les valeurs a_{ij} que p_j reçoit des autres joueurs honnêtes correspondent aux valeurs b_{ij} que p_j avait reçues de D.

5. Le joueur p_j compare la valeur a_{ij} reçue par p_i à la valeur b_{ij} reçues par D, pour $1 \leq i \leq N$. Si pour un index (i, j) il y a une incohérance entre les deux valeurs, p_j dévoile publiquement l'index (i, j), D doit alors dévoiler publiquement la valeur a_{ij}.

   Si p_j détecte plus de t incohérances, il demande publiquement à D de dévoiler toutes les valeurs qu'il lui a données (les valeurs interpolant à $f_j(y)$ et $g_j(x)$) et vote pour la disqualification de D.

6. Si un joueur p_k observe une valeur publique incohérante avec ce qu'il possède, il demande à D de dévoiler publiquement les valeurs qu'il lui a données et vote pour la disqualification de D.

7. Finalement, chaque joueur p_i observe toutes les valeurs publiques et privées que D a données, s'il y a une incohérance, p_i demande à D de dévoiler les valeurs qu'il lui a données et vote pour le disqualifier.

8. S'il y a eu $t + 1$ joueurs différents (ou plus) qui ont voté pour la disqualification de D, ou si D a refusé de publier une valeur demandée, D est clairement en faute et chaque joueur p_i se doit de retourner $ver_i = 0$.

Rec(D, P, s) :

1. Chaque participant p_i dévoile publiquement ses valeurs $a_{1i}, \ldots, a_{Ni}$.
2. Interpoler les valeurs de p_i en la fonction $g_i(x)$ et calculer $g_i(0)$, pour chaque i.
3. Utiliser le decodeur de Berlekamp-Welch (voir, par exemple, [BW] pour un tel algorithme).

♠

Théorème 2.3.2 (PSV de Feldman) *Le PSV de Feldman est un PSV* $(N/3, N)-$*robuste.*

Preuve: [Sécurité du PSV de Feldman] Nous démontrons que chaque critère d'un PSV est satisfait (se référer à la section 2.2).

- **Cohérence**
  - Le calcul de ver_i se fait à la fin de l'étape 8, suite à une observation des données publiques (ce qui, dans un modèle de Turing, consiste à observer l'état du ruban de communication partagé). Tous les joueurs honnêtes observeront la même chose et calculeront donc la même valeur ver.
  - L'item précédent étant prouvé, il suffit de montrer qu'un joueur honnête p_H ne votera jamais pour la disqualification d'un initiateur honnête D_H.

    Les votes de disqualification de D se font aux étapes 3, 5, 6 et 7. Examinons ce qui se passe en ces étapes du point de vue de p_H lorsque l'initiateur est honnête (D_H).

    À l'étape 3, p_H ne votera jamais pour la disqualification de D_H puisqu'il reçoit des valeurs interpolant bien en des polynômes de degrés au plus t (puisque D_H est honnête).

À l'étape 5, p_H reçoit au plus t valeurs incohérantes (des joueurs malhonnêtes) si D est honnête. Il ne votera donc jamais pour la disqualification de D_H.

Aux étapes 6 et 7, un initiateur honnête publiera toujours des valeurs cohérentes, donc p_H ne votera jamais pour la disqualification de D_H à ces étapes-là.

En conclusion, si D_H est honnête, il y aura au plus t joueurs qui voteront pour sa disqualification (les joueurs malhonnêtes) et p_H calculera donc $ver_H = 1$.

- **Confidentialité**
  - Notons d'abord que si l'initiateur est malhonnête, nous ne sommes pas dans l'obligation de protéger la confidentialité de son information (rien ne l'empêche de dévoiler son secret à qui il veut !). Si l'initiateur est honnête, il suffit de remarquer qu'un adversaire n'apprend rien de plus au cours des étapes 2 à 8 que ce qui lui a été dit à l'étape 1. Le critère de confidentialité est alors une conclusion directe des propriétés d'un polynôme bivarié de degré au plus t.
- **Reconstitution**
  - Ceci est une conséquence directe de l'utilisation des correcteurs de polynômes (voir la discussion de la section **??**) et des propriétés du polynôme bivarié.

□

Il peut être intéressant de se demander s'il est possible de tolérer un plus grand nombre de participants malhonnêtes dans le PSV de Feldman, en ne modifiant que le protocole *Rec*. La réponse à cela est négative et elle est donnée par la construction suivante :

Soit $N = 3$, nous considérons 3 participants, p_1, p_2 et l'initiateur $D = p_3$. Disons que p_1 et p_2 sont honnêtes (D sera malhonnête). Nous allons montrer comment

D peut distribuer des parts de façon à ce que p_1 et p_2 ne se rendent pas compte d'une incohérence durant la phase $Distr$, mais que dans la phase Rec D peut imposer la reconstitution d'une de deux valeurs différentes, s_1, s_2, à son choix.
Considérons le corps $K = F_5$ et l'élément primitif $w = 2 \in K$ et disons que les points ω^0 et ω^1 de K sont utilisé pour la mise en partage du secret. La tâche d'un initiateur D malhonnête, voulant mettre en partage deux secrets, consiste dans ce cas-ci à trouver deux polynômes bivariés de degré 1, $f(x, y)$ et $g(x, y)$, partageant respectivement deux secrets distincts s_1 et $s_2 \in K$.
Notons les polynômes par

$$f(x, y) = s_1 + a_1 x + b_1 y$$
$$et \quad g(x, y) = s_2 + a_2 x + b_2 y$$

Puisque dans le protocole $Dist$ p_1 et p_2 vont comparer, entre-eux, les valeurs aux points (w^0, w^1) et (w^1, w^0), ces fonctions sont soumises aux contraintes suivantes :

$$f(w^0, w^1) = g(w^0, w^1),\ f(w^1, w^0) = g(w^1, w^0)\ et\ s_1 \neq s_2$$

Il faut donc trouver $s_1 \neq s_2$, a_1, b_1, a_2, b_2 qui satisfont le système d'équations

$$\begin{aligned} s_1 + a_1 w^1 + b_2 w^0 &= s_2 + a_2 w^1 + b_2 w^0 \\ s_1 + a_1 w^0 + b_1 w^1 &= s_2 + a_2 w^0 + b_2 w^1 \\ et\ s_1 - s_2 \neq 0 & \end{aligned}$$

En posant $\Delta s := s_1 - s_2$, $\Delta a = a_1 - a_2$ et $\Delta b = b_1 - b_2$, et sachant que $w^0 = 2^0 \ mod\ 5 = 1$ et que $w^1 = 2^1 \ mod\ 5 = 2$, nous nous retrouvons à résoudre le système

$$\begin{aligned} \Delta s + 2 \cdot \Delta a + \Delta b &= 0 \\ \Delta s + \Delta a + 2 \cdot \Delta b &= 0 \\ et\ \Delta s \neq 0 & \end{aligned}$$

Cependant, ce système a plusieurs solutions (nous avons 2 équations avec 3 inconnus et une contrainte $\Delta s \neq 0$; pour toute valeur Δs il existe une solution).

L'initiateur peut choisir Δs pour satisfaire son choix de s_1 et s_2.
Supposons que l'initiateur veut mettre en partage $s_1 = 3$ et $s_2 = 1$, une solution possible est alors de prendre

$$\Delta s = 2, \Delta a = 1 \; et \; \Delta b = 1.$$

Nous pouvons donc prendre, par exemple,

$$s_1 = 3, s_2 = 1, a_1 = 2, a_2 = 1, b_1 = 2 \; et \; b_2 = 1$$

Ce qui fait que l'on se retrouve avec les fonctions

$$f(x, y) = 3 + 2x + 2y$$

et

$$g(x, y) = 1 + x + y.$$

Les matrices $[f(w^{i-1}, w^{j-1})]_{1 \leq i \leq 3, 1 \leq j \leq 3}$ et $[g(w^{i-1}, w^{j-1})]_{1 \leq i \leq 3, 1 \leq j \leq 3}$ sont respectivement

$$\begin{pmatrix} 2 & 4 & 3 \\ 4 & 1 & 0 \\ 3 & 0 & 4 \end{pmatrix} \text{ et } \begin{pmatrix} 3 & 4 & 1 \\ 4 & 0 & 2 \\ 1 & 2 & 4 \end{pmatrix}$$

Nous constatons que les valeurs de f et de g au point (w^0, w^1) correspondent, il va de même pour les valeurs de f et de g au point (w^1, w^0) . D pourrait donc distribuer les points $(a_{11} = 2, a_{12} = 4, a_{13} = 3)$ et $(b_{11} = 2, b_{21} = 4, b_{31} = 3)$ (première ligne et colonne de $[f(w^{i-1}, w^{j-1})]_{1 \leq i \leq 3, 1 \leq j \leq 3}$) au joueur p_1 et les points $(a_{21} = 4, a_{22} = 0, a_{23} = 2)$ et $(b_{12} = 4, b_{22} = 0, b_{32} = 2)$ (deuxième ligne et colonne de $[g(w^{i-1}, w^{j-1})]_{1 \leq i \leq 3, 1 \leq j \leq 3}$) à p_2 . p_1 et p_2 ne retrouveront aucune incohérence entre leurs valeurs durant $Dist$. Lorsque p_1 compare ses valeurs avec D, ce dernier choisit les valeurs correspondantes de $[f(w^{i-1}, w^{j-1})]_{1 \leq i \leq 3, 1 \leq j \leq 3}$, et lorsque p_2 compare ses valeurs avec D, D choisit les valeurs correspondantes de $[g(w^{i-1}, w^{j-1})]_{1 \leq i \leq 3, 1 \leq j \leq 3}$. p_1 et p_2 ne retrouveront donc aucune incohérence et ils calculeront chacun $ver = 1$.
Lors de la phase de reconstitution, D peut choisir de publier ses valeurs (a_{31}, a_{32}, a_{33})

de $[f(w^{i-1}, w^{j-1})]_{1\leq i\leq 3, 1\leq j\leq 3}$ ou de $[g(w^{i-1}, w^{j-1})]_{1\leq i\leq 3, 1\leq j\leq 3}$, dépendant de s'il veut que la valeur s_1 ou la valeur s_2 soit reconstituée. Dans les deux cas, seuls p_1 ou p_2 (jamais les deux en même temps) pourront se plaindre, et les valeurs de D seront totalement cohérentes avec un des deux joueurs. Puisque nous considérons que 2 joueurs sont honnêtes, D passera pour honnête, alors qu'un de p_1 ou p_2 sera considéré comme étant malhonnête. Le point fondamental à remarquer, peu importe la méthode de reconstitution du secret, est que les participants qui sont honnêtes (voir ici p_1 et p_2) ont tous calculé $ver = 1$, mais il n'y a pas eu de valeur fixée (σ) à la fin de *Dist* telle qu'il y ait seuleument une probabilité exponentiellement petite (2^{-k}) que les participants honnêtes ne calculent pas σ à la fin de *Rec* (D peut fixer σ à une de deux valeurs différente après la phase *Rec*). Le protocole viole donc le critère de Reconstitution d'un PSV (voir section 2.2).

Heureusement, il existe cependant une façon de modifier le protocole *Dist* du PSV de Feldman de sorte à pouvoir construire un PSV $(N/2, N) - robuste$. Il suffit de demander à chaque participant de calculer une signature digitale pour chacun de leurs éléments. Lorsqu'un joueur p_j compare une valeur a_{ij} avec un joueur p_i, nous demandons à p_i de fournir une signature digitale de a_{ij}. Cette signature peut alors être utilisée par p_j durant la phase *Rec* pour contester une mauvaise part de p_i. Observer que dans notre exemple plus haut, si l'initiateur avait signé toutes ses parts, p_1 ou p_2 auraient pu contester les parts incohérentes de D, lors de *Rec*, en publiant la signature de D qui authentifie une autre valeur pour le même point. Deux signatures digitales d'un joueur p, pour deux valeurs distinctes d'un même point de la fonction bivariée indiquent clairement une tricherie de la part de p. Cette idée a en fait été proposée pour la première fois par Micali et Feldman même ([FM88]), en introduisant l'utilisation de signatures digitales à sécurité cryptographique. Par contre, le problème est que nous cherchons une sécurité *inconditionnelle* (basée sur aucune hypothèspas σ à la fin de *Rec* e cryptographique). Heureusement un schéma, présenté pour la première fois par Rabin et Ben-Or dans [RB89] (et par la suite raffiné et utilisé pour ses propriétés d'authentification dans [CDD$^+$99]), nous porte secours. Ce schéma, appelé protocole de vérification d'information (VI) est le sujet de la prochaine section.

2.4 Protocoles de vérification d'information (VI)

Dans un protocole de vérification d'information (VI) nous avons un initiateur D qui confie un secret s à un intermédiaire I, ce dernier possède le rôle de dévoiler le secret à un receveur R à un temps ultérieur. Le protocole permet à I de se convaincre que R acceptera bien s en tant que secret provenant de D (pour contrer un D malhonnête) et permet aussi a R, une fois le secret reçu, de se convaincre qu'il vient bien de D (pour contrer un I malhonnête). Prenez note par contre qu'un protocole de VI n'est pas équivalent à une signature digitale, mais possède les propriétés d'authentification dont nous avons besoin pour construire un PSV $(N/2) - robuste$ et il ne dépend d'aucune hypothèse cryptographique. Les protocoles de VI ont une probabilité d'erreur, laquelle est incarnée par les PSV $(N/2, N) - robuste$ que nous présenterons. Par contre cela n'est pas du tout grave puisque la probabilité d'erreur peut être rendue exponentiellement petite et que de toute façon, d'après les résultats de [RB89], nous savons qu'il est impossible qu'un PSV tolérant $t \geq N/3$ participants malhonnêtes soient sans probabilité d'erreur.

Formellement, un protocole VI consiste en un triplet de protocoles $Distr$, $AuthVal$, $DevoilerVal$, où

- $Distr(D, I, R, s)$ est initié par I. Dans cette phase, D donne le secret s à l'intermédiaire I ainsi que d'autres données auxiliaires à I et à R.
- $AuthVal(I, R, s)$ est initié par I. Dans cette phase, I s'assure que R (si honnête) acceptera le s qu'il possède.
- $DevoilerVal(I, R, s)$ est initié par I. Dans cette phase, R reçoit la valeur s

de I ainsi que des données auxiliaires et décide d'accepter ou de rejeter s.

Les protocoles doivent vérifier les critères suivants :

- **Exactitude**
  - Si D et R sont honnêtes, R acceptera toujours la valeur s donnée par I si elle provient vraiment de D et rejettera avec probabilité $\geq 1 - \frac{1}{2^k}$ toute autre valeur s'.
  - Si I et R sont honnêtes, I saura avec probabilité d'erreur $\leq \frac{1}{2^k}$, après les phases $Distr$ et $AuthVal$, si la valeur s qu'il possède sera acceptée par R.
- **Confidentialité**
  - Si D et I sont honnêtes, alors R n'apprendra aucune information (au sens de Shannon) à propos de s en autant que la phase $DevoilerVal$ ne soit pas encore commencée.

2.5 Vérification d'information (VI) de Rabin et Ben-Or

Nous présentons le premier protocole VI, de Rabin et Ben-Or, qui a permis de construire un protocole de PSV $(N/2, N) - robuste$ inconditionnellement sûr. Nous travaillons toujours dans un corps $K = GF(q)$ où $q > 2^k$, k étant le paramètre de sécurité.

Protocole[VI de Rabin et Ben-Or]

Distr(D, I, R, s) :

- L'initiateur D choisit $2k$ nombres aléatoires $y_1, \ldots y_{2k} \in_R K$ et $2k$ nombres aléatoires $b_1, \ldots, b_{2k} \in_R K^*$.

- D calcule $c_i \leftarrow s + b_i y_i$, pour $1 \leq i \leq 2k$.
- D donne $(s, y_1), \ldots, (s, y_{2k})$ à I.
- D donne $(b_1, c_1), \ldots, (b_{2k}, c_{2k})$ à R.

AuthVal(I, R, s) :

- I choisit k indices aléatoires distincts $1 \leq i_1, \ldots, i_k \leq 2k$ et les publie.
- R dévoile les paires $(b_{i_1}, c_{i_1}) \ldots, (b_{i_k}, c_{i_k})$.
- Pour chacun des indices choisis $i_j,\ 1 \leq j \leq k,\ I$ vérifie si

$$s + b_{i_j} y_{i_j} = c_{i_j}$$

- Si toutes les paires satisfont l'égalité, I conclut que R (si honnête) acceptera s, sinon il conclut que R le rejettera.

DevoileVal(I, R, s) :

- I donne s et $(y_1, \ldots, y_{2k})$ à R.
- Pour chaque indice i_j dont les paires (b_{i_j}, c_{i_j}) n'ont pas été dévoilées, R vérifie si $s + b_{i_j} y_{i_j} = c_{i_j}$.
- Si l'égalité est vérifiée pour *au moins une paire*, R accepte la valeur.

♠

Preuve: [Sécurité du protocole de VI de Rabin- Ben-Or] Nous démontrons que chaque critère d'un protocole VI est satisfait (se référer à la section 2.4).

- **Exactitude**
  - Nous supposons que D et R sont honnêtes et que le secret de D est s. Si I donne la valeur s qui provient vraiment de D, il est clair que R l'acceptera.

    Observons donc ce qui se passe lorsque I veut trouver une valeur $s' \neq s$ pour donner à R.

    I veut donc essayer de trouver y' tel que l'égalité $s' + by' = c$ recherchée par R soit satisfaite. Notons d'abord que I n'obtient aucune information par rapport à b durant les phases $Distr$ et $AuthVal$, puisque la distribution de b est indépendante des autres valeurs.

    Maintenant, puisque $s + by = c$, nous savons que $s' + by' = c$ si et seulement si $s - s' + b(y - y') = 0$. Ce qui veut dire que *si* I est capable de trouver s' et y' satisfaisant l'égalité que R recherche, *alors* il est capable de résoudre $s - s' + b(y - y') = 0$ pour retrouver b. Vu que I n'a aucune information à propos de b, sa meilleure façon de faire est de choisir y' uniformément au hasard dans K. La probabilité de succès de I est donc de $1/|K| \leq 2^{-k}$.
  - Nous supposons ici que I et R sont honnêtes.

    Si D est honnête, R acceptera toujours s. Si D est malhonnête et qu'il a donné à I une valeur s que R n'acceptera pas, sans que I s'en aperçoive durant $AuthVal$, cela voudrait dire que I a eu la malchance de choisir les k paires (parmi les $2k$) qui ne vérifient pas l'égalité que R recherche. Nous aurions donc

    $$Prob(I\ pense\ que\ R\ acceptera\ s \mid R\ rejette\ s) = \begin{pmatrix} 2k \\ k \end{pmatrix}^{-1}$$

    puisque I choisit ses indices de façon uniformément aléatoire. Il suffit alors de se rappeler de la proposition suivante :

$$\binom{2k}{k} = \sum_{i=0}^{k} \binom{k}{i}^2$$

(voir par exemple [Spi94] pour une preuve)

et de noter que

$$\begin{aligned}
\binom{2k}{k}^{-1} &= \frac{1}{\sum_{i=0}^{k} \binom{k}{i}^2} \\
&\leq \frac{1}{\sum_{i=0}^{k} \binom{k}{i}} \\
&= \frac{1}{2^k} \qquad \texttt{(binôme de Newton)}
\end{aligned}$$

Ceci prouve que la malchance en question n'arrive qu'avec probabilité $\leq \frac{1}{2^k}$.

- **Confidentialité**
  - R n'a aucune information à propos de s avant la phase $DevoileVal$ puisque pour des valeurs b et c fixes, toute valeur de s est telle qu'il existe un y unique satisfaisant $c = s + by$ et les distributions de s et de y sont indépendantes de celles de b et de c.

$\square$

2.6 Vérification d'information (VI) de CDDHR

Presque dix ans après l'introduction des protocoles de VI, Cramer, Damgård, Dziembowski, Hirt et Rabin ont présenté dans [CDD+99] un nouveau protocole de VI. Ce dernier protocole ne demande qu'une communication de complexité logarithmique par rapport au paramètre de sécurité k (au lieu de linéaire, comme pour celui de Rabin et Ben-Or). De plus, le nouveau protocole possède une propriété additionnelle de linéarité, dont nous discuterons après la description de l'algorithme.

Pour présenter le protocole, nous avons besoin d'une nouvelle définition (donnée dans [CDD+99])

Définition: 2.6.1 *Un vecteur* $(x_1, x_2, x_3) \in K^3$ *est dit* $1_\alpha-$*cohérant s'il existe un polynôme p de degré 1, sur* K*, tel que* $p(0) = x_1,\ p(1) = x_2$ *et* $p(\alpha) = x_3$*.* ♣

Protocole[VI de CDDHR]

Distr(D, I, R, s) :

1. D choisit une valeur $\alpha \in_R K\backslash\{0, 1\}$ ainsi que deux autres valeurs aléatoires, $y, z \in_R \{K^2 \mid (s, y, z)$ soit $1_\alpha-$*cohérant*$\}$. D dévoile α publiquement.
2. D choisit un vecteur aléatoire (s', y', z') qui est $1_\alpha-$*cohérant.*
3. D donne s, s', y, y' à I.
4. D donne z, z' à R.

AuthVal(D, I, R, s) :

1. I choisit un élément aléatoire $d \in_R K$ et publie $d, (s' + ds)$ *et* $(y' + dy)$.
2. Si D observe la publication d'une mauvaise valeur, il publie s et y. R calcule alors z de façon à ce que (s, y, z) soit $1_\alpha-$*cohérant* et le protocole se termine.
3. R vérifie si

$$(s' + ds, y' + dy, z' + dz)$$

est $1_\alpha-$*cohérant*, il publie **accepte** ou **rejette** selon le cas.

4. Si D observe que R est malhonnête, il le déclare publiquement et publie z et s. I ajuste y si nécessaire de sorte que (s, y, z) soit $1_\alpha-$*cohérant*, et le protocole se termine.
5. Si D n'a pas déclaré R comme étant malhonnête en 4., mais que R a publié **rejette** en 3., D doit publier s et y et ensuite R doit alors ajuster sa valeur z de sorte que (s, y, z) soit $1_\alpha-$*cohérant*.

DevoileVal(I, R, s) :

1. I publie (s, y).
2. R vérifie que (s, y, z) est bien $1_\alpha-$*cohérant* et **accepte** ou **rejette** de façon appropriée.

♠ Le vecteur (s, y, z) est appelé vecteur de vérification.

Nous pouvons maintenant remarquer que si deux vecteurs (possiblement différents) (x, y, z) et (x', y', z') sont $1_\alpha-$*cohérants*, alors toute combinaison linéaire de ces deux vecteurs sera aussi $1_\alpha-$*cohérante*. Cela implique que si nous avons deux secrets s_1, s_2 et l'habilité de les vérifier, nous pouvons aussi vérifier toute combinaison linéaire de s_1 et s_2. Cette propriété sera utilisée dans les PSV, ainsi que dans les preuves de validité du protocole VI de CDDHR.

Preuve: [Sécurité du protocole de VI de CDDHR] Nous démontrons que chaque critère d'un protocole VI est satisfait (se référer à la section 2.4).

- **Exactitude**
  - Nous supposons ici que D et R sont honnêtes et que le secret de D est s.

    Il est clair que si I donne à R la valeur s que D lui a vraiment donnée, R acceptera.

    Nous étudions donc le cas où I veut donner une valeur $s' \neq s$.

    Remarquons d'abord que I n'apprend aucune information à propos de α durant *Distr* et *AuthVal* : ce que I reçoit durant *Distr* a une distribution indépendante de α. Durant $AuthVal$, si I envoie les bonnes valeurs, il connaît d'avance la réponse de R. S'il n'envoie pas les bonnes valeurs, D se plaindra et le protocole termine. Donc I n'obtient aucune information à propos de α.

Notons que ces arguments sont vrais peu importe le nombre de fois que $AuthVal$ a été éxécuté (ceci est important dans le cas ou nous avons plusieurs receveurs).

Maintenant, lors de la phase $DevoileVal$, si I envoie $s' \neq s, y' \neq y$, R acceptera seulement si (s', y', z) est $1_\alpha-$*cohérant*. Vu que (s, y, z) est $1_\alpha-$*cohérant*, (s', y', α) est en fait $1_\alpha-$*cohérant* si et seulement si $(s - s', y - y', 0)$ l'est (par la propriété de linéarité). Ceci nous donne une équation non triviale que I pourrait résoudre pour retrouver α. I peut faire ceci avec probabilité au plus $1/(|K| - 2|)$.

D'un autre côté, si I envoit des valeurs s', y' telles que R n'accepte pas, tout ce que I apprend c'est que la droite passant par les points $(0, s')$ et $(1, y')$ n'est pas la droite définit par les points que R possède. R peut alors éliminer s' comme solution possible.

Par déduction, nous pouvons voir que si au plus l valeurs peuvent êtres éliminées du à des réponses négatives de R, au moins $|K| - l - 2$ candidats pour α resteront valides. Il s'ensuit que I peut deviner s avec probabilité au plus $1/(|K| - l - 2)$.

Lorsque nous utiliserons ce protocole de VI pour construire notre prochain PSV, l sera linéaire en n. La probabilité d'erreur (la probabilité que I puisse authentifier une mauvaise valeur) sera donc d'au plus $1/(|K| - O(n) - 2) = 1/(2^k - O(n)) = 2^{-k+O(lg\, n)}$.

- Nous supposons ici que I et R sont honnêtes.

  Si D publie les valeurs s et y ou z et α, la propriété est trivialement satisfaite.

  Nous observons donc la situation lorsque D envoie initialement à I des valeurs incohérantes et que R calcule **accepte**, sans plainte de D, à la phase $AuthVal$. Notons que s'il existe au moins deux valeurs $d_1 \neq d_2$ telles que $(s' + d_1 s, y' + d_1 y, z' + d_1 z)$ et $(s' + d_2 s, y' + d_2 y, z' + d_2 z)$ sont tous les deux $1_\alpha -$*cohérants*, alors par la propriété de linéarité (s, y, z) l'est aussi. Ceci nous montre par contraposé que si (s, y, z) n'est pas $1_\alpha -$*cohérant*, il ne peut exister qu'une seule valeur $d \in K$ telle que $(s' + ds, y' + dy, z' + dz)$ soit $1_\alpha -$*cohérant*. Il y a donc au plus une valeur d que R acceptera durant la phase $AuthVal$. Puisque I choisit d de façon uniformément aléatoire dans K, la probabilité de succès de tricherie de D est donc de $1/|K|$.

- **Confidentialité**
  - Nous supposons ici que D et I sont honnêtes.

    Durant la phase $AuthVal$, R peut soit publier **accepte** ou **rejette**. Puisque D et I sont honnêtes, si R rejette D dévoile z et α et ensuite termine le protocole, nous ne sommes donc plus obligé de dissimuler s de R. Si par contre R accepte, tout ce qu'il peut apprendre durant $AuthVal$ ce sont les valeurs $z, z', d, s' + ds, y' + dy$, provenant d'une distribution indépendante de s. Il s'ensuit donc que R n'apprend rien à propos de s.

$\square$

2.7 Signature-VI

Lorsqu'une personne reçoit une signature digitale d'un signataire, cette première peut montrer cette signature ainsi que sa validité à n'importe qui. Cette propriété d'authentification peut être obtenue à l'aide des protocoles de VI et sera utilisée lors de la construction du prochain PSV que nous présentons. Une *Signature-VI*, dans le contexte des PSV, se fait comme suit : *Distr* est exécuté pour une valeur s par D avec un intermédiaire I et les receveurs $p_1, \cdots, p_N$ (les participants du PSV). Ensuite, $AuthVal$ est exécuté par I avec chaque joueur $p_i \in P$. Plus tard, si I veut authentifier aux participants de P que s provient bien de D, il exécute $DevoileVal$ avec tous les participants. Si au moins $t+1$ participants acceptent la valeur s, nous dirons que la signature-VI a été confirmée. Nous dénoterons une telle signature par $\sigma_s(D, I)$ (le premier argument spécifie l'initiateur et le deuxième specifie l'intermédiaire).

2.8 Partage de secret vérifiable (PSV) de CD-DHR

Nous sommes maintenant aptes à présenter le PSV de [CDD+99]. Ce protocole est basé sur la solution de Feldman, omettant la nécessité de codes correcteurs. Voici la définition d'un terme qui nous sera utile :

Définition: 2.8.1 *Un vecteur* $(e_1, \ldots, e_N) \in K^N$ *est un vecteur* **t-cohérant** *s'il existe un polynôme* p *de degré au plus* t *tel que* $p(i) = e_i$ *pour* $1 \leq i \leq N$. ♣

Le protocole de partage de secret vérifiable se définit comme suit

Protocole[PSV de CDDHR]

Distr(D, P, s) :

1. D choisit un polynôme bivarié $f(x,y)$ aléatoire de degré au plus t en chaque variable, tel que $f(0,0) = s$. Nous construisons la matrice

$$\begin{pmatrix} s_{11} & s_{12} & \dots & s_{1i} & \dots & s_{1N} \\ & & & \vdots & & \\ & & & \vdots & & \\ s_{i1} & s_{i2} & \dots & s_{ii} & \dots & s_{iN} \\ & & & \vdots & & \\ s_{N1} & s_{N2} & \dots & s_{Ni} & \dots & s_{NN} \end{pmatrix}$$

   où $s_{ij} = f(i,j)$. La rangée i définit $f_i(y)$ et la colonne i définit $g_i(x)$. Les $g_i(0)$ (qui ne sont pas représentés dans la matrice) sont les coefficients qui peuvent être interpolés pour retrouver s.

2. D donne $a_{i1} = s_{i1}, \dots, a_{iN} = s_{iN}$ et $b_{1i} = s_{1i}, \dots, b_{Ni} = s_{Ni}$ au joueur p_i, pour $i = 1, \dots, N$. Pour chaque valeur a_{ij}, b_{ji}, D attache une signature-VI $\sigma_{a_{ij}}(D, p_i)$, $\sigma_{b_{ji}}(D, p_i)$

3. Le joueur p_i vérifie que les deux ensembles $a_{i1}, \dots, a_{iN}$ et $b_{1i}, \dots, b_{Ni}$ obtenus de D sont bien *t-cohérant*.

   Si cela n'est pas le cas, p_i publie ses valeurs ainsi que les signatures-VI de D s'y rapportant. Si un joueur observe la publication de valeurs incohérantes avec des signatures-VI valides de D, il vote pour la disqualification de D.

4. Chaque p_i donne $a_{ij} = f(i,j)$ au joueur p_j ainsi qu'une signature-VI, $\sigma_{a_{ij}}(p_i, p_j)$, qu'il génére pour cette valeur.

   Noter que si l'initiateur est honnête, les valeurs a_{ij} que p_j reçoit des autres joueurs honnêtes correspondent aux valeurs b_{ij} que p_j avait reçues de D.

5. Le joueur p_j compare la valeur a_{ij} reçue par p_i à la valeur b_{ij} reçue par D, pour $1 \leq i \leq N$. Si pour un index (i,j) il y a une incohérance entre les deux valeurs (ou s'il n'a pas reçu la valeur ou la signature de p_i), p_j dévoile publiquement a_{ij} et $\sigma_{a_{ij}}(D, p_j)$.

6. Si un joueur p_i observe une valeur publique a_{ij}, $\sigma_{a_{ij}}(D, p_j)$, incohérante avec ce qu'il possède, il publie sa valeur a_{ij} et $\sigma_{a_{ij}}(D, p_i)$.
7. Finalement, chaque joueur p_i observe toutes les valeurs publiques. S'il observe pour un index (j, k) deux différentes signatures-VI (associées à deux valeurs distinctes) valides de la part de D, p_i vote pour la disqualification de D.
8. S'il a eu $t + 1$ joueurs différents, ou plus, qui ont voté pour la disqualification de D, on demande à chaque joueur p_i de retourner $ver_i = 0$.

Rec(D, P, s) :

1. Chaque joueur p_i publie ses valeurs $a_{1i}, \ldots a_{Ni}$ avec les signature-VI pour les valeurs a_{ij} recues des joueurs p_j.
2. Chaque joueur p_i vérifie si les valeurs de p_j sont *t-cohérantes* et si les signatures sont valides. S'il y a une incohérance, p_j est disqualifié.
3. Les valeurs des participants non disqualifiés sont utilisées pour interpoler le secret.

♠

Théorème 2.8.1 *Le PSV de CDDHR est* $(t + 1, 2t)-$*robuste.*

Preuve: Nous démontrons que chaque critère d'un PSV est satisfait (se référer à la section 2.2).

- **Cohérence**
  - Le calcul de ver_i se fait à la fin de l'étape 8, suite à une observation des données publiques. Tous les joueurs honnêtes observeront la même chose et calculeront donc la même valeur ver.
  - Il est facile de constater que si D est honnête, jamais il ne signera une valeur incohérante et par conséquence ne se fera donc jamais disqualifier.

- **Confidentialité**
  - Si l'initiateur est malhonnête, nous ne sommes pas dans l'obligation de protéger la confidentialité de son information. Si par contre l'initiateur est honnête, il suffit de remarquer qu'un adversaire n'apprend rien de plus au courant des étapes 2 à 8 que ce qui lui a été dit à l'étape 1. Le critère de confidentialité est alors une conclusion directe des propriétés d'un polynôme bivarié de degré au plus t.
- **Reconstitution**
  - Nous montrons d'abord qu'une valeur fixe est définie par la distribution des joueurs honnêtes, peu importe si l'initiateur est honnête ou non. Définissons r comme étant le secret qui interpole à travers les parts que possèdent les $t+1$ premiers participants honnêtes. Leurs parts sont *t-cohérants* (ils se seront plaints autrement) et avec probabilité au moins $1 - O(2^{-k+lg\,n})$, ils possèdent aussi des signatures valides pour ces parts, ce qui indique que les parts reposent sur un unique polynôme $f'(x, y)$ tel que $f'(0, 0) = r$ (une constante fixe quelconque). Observons le point de vue d'un autre participant honnête qui n'est pas inclu dans ce dernier ensemble. Ce participant a vérifié ses parts avec les $t+1$ premiers joueurs honnêtes. De plus, il a aussi vérifié les signatures. Les parts de ce participant (qui sont *t-cohérants*) s'ajoutent donc aux autres et l'ensembe des points reste *t-cohérants*, ces points définissent donc la même fonction $f'(x, y)$.

    Maintenant, si D est honnête, il est évident de voir que $r = s$.

    Une valeur différente de r ne peut être interpolée que lorsqu'un joueur malhonnête a pu produire un ensemble de n valeurs, *t-cohérant*, avec une signature valide pour chacune de ces valeurs. Il est clair qu'au moins $t+1$ de ces signatures proviennent de joueurs honnêtes. Nous avons montré que les parts des joueurs honnêtes reposent sur le polynôme $f'(x, y)$, donc si les parts du joueur malhonnête sont *t-incohérants* (le joueur est disqualifié durant *Rec* si ce n'est pas le cas), ces points reposent sur la même fonction. L'adversaire ne peut donc pas influencer le secret qui sera révélé.

□

Chapitre 3

Protocoles de Calculs Multipartites

Dans ce chapitre, nous présentons les protocoles de calculs multipartites (PCM) inconditionnellement sûrs. Ceux-ci utilisent les protocoles de PSV que nous avons vus au chapitre précédent, ainsi que d'autres schémas que nous présenterons. Dans un PCM, nous avons plusieurs individus $p_1, \ldots p_N$, ayant chacun une donnée privée $x_i \in K$, $1 \leq i \leq N$, qui veulent calculer la valeur d'une fonction publique F au point $(x_1, \ldots, x_N)$.

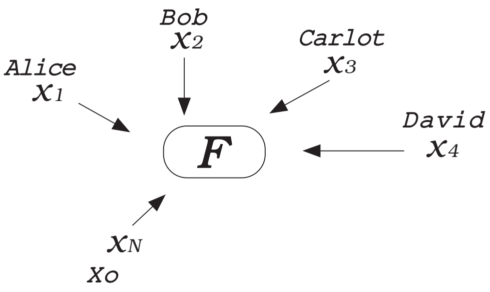

Fig. 3.1: Calcul multipartite

Goldreich, Micali et Wigderson ont prouvé que pour réaliser un PCM, il suffit de toujours faire en sorte que la valeur de $F(x_1, \ldots, x_N)$ soit révélée publiquement et que F soit donnée par un circuit arithmétique (i.e. un circuit composé de portes d'addition et de multiplication) de K^N à K, où K est un corps fini. Pour arriver à satisfaire les critères d'un PCM, chaque joueur distribue sa donnée privée aux autres participants à l'aide d'un PSV. Par la suite, les joueurs exécutent des protocoles d'addition, de multiplication par une constante et de multiplication de deux secrets en utilisant les parts qu'ils possèdent de ces secrets. Il faut implanter ceci de façon à ce que les autres joueurs soient convaincus que les calculs sont exacts sans qu'ils puissent pour autant apprendre de l'information à propos des parts utilisées comme opérandes ni à propos du résultat. Le résultat final du calcul de la fonction se retrouve sous une forme distribuée et peut donc être révélé par le protocole $DevoileVal$ du PSV utilisé. Nous présentons donc un PCM hybride s'inspirant de [BGW88], [CCD88], [Bea90], [SS98] et [CDD$^+$99]. Dans ce PCM, les protocoles d'addition et de multiplication par une constante sont simples. Le protocole de multiplication est un peu plus futé et demande, entre autres, une preuve à divulgation nulle. Nous présentons ces protocoles, ils sont tous construits à l'aide du PSV de CDDHR (section 2.8) qui se sert du protocole de VI de CDDHR (section 2.6).

3.1 Forme d'un secret mis en partage de façon vérifiable

Pour toutes les opérations d'addition, de multiplication par une constante et de mutiplication de deux secrets que nous allons présenter, le résultat doit se retrouver sous la forme d'un secret mis en partage de façon vérifiable. Puisque le PSV utilisé est celui de CDDHR, un résultat r doit donc se retrouver partagé par des parts $r_1, \ldots, r_N$, où r_i appartient au joueur p_i qui possède les sous-parts $r_{1i}, \ldots, r_{Ni}$, où finalement r_{ji} est aussi connu par le joueur p_j et p_i possède une signature-VI de ce dernier pour cette valeur. De même, p_i connait les valeurs r_{ij} de p_j, pour $1 \leq j \leq N$.

3.2 Addition multipartite

Nous voulons calculer $u+v$ de sorte à ce que le résultat soit un secret vérifiable. Disons que u est mise en partage de façon vérifiable par le polynôme $f^u(x,y)$ et que v est mise en partage par le polynôme $f^v(x,y)$. Pour arriver au résultat voulu, chaque joueur p_i calcule la somme de chacun de ses parts de f^u et f^v, $u_{ji}+v_{ji}$, $1 \leq j \leq N$, en privé. Il est clair que $u_{ji}+v_{ji}$ est une valeur sur le polynôme $f^{u+v} := f^u + f^v$ qui a comme terme constant $u+v$.

$$\begin{pmatrix} f^u(1,1)+f^v(1,1) & \dots & f^u(1,N)+f^v(1,N) \\ & \vdots & \\ & f^u(i,j)+f^v(i,j) & \\ & \vdots & \\ f^u(N,1)+f^v(N,1) & \dots & f^u(N,N)+f^v(N,N) \end{pmatrix}$$

Ensuite, pour que $u+v$ soit un secret vérifiable, chaque p_i doit posséder une signature-VI pour les sous-parts $u_{1i}+v_{1i}, \dots, u_{Ni}+v_{Ni}$. Puisque le IC de CDDHR est linéaire, il suffit d'additionner les deux signatures-VI de chacune de ces valeurs. En d'autre mots, si le vecteur de vérification de u_{ji} est $(u_{ji}, y_{u_{ji}}, z_{u_{ji}})$ et que celui de v_{ji} est $(v_{ji}, y_{v_{ji}}, z_{v_{ji}})$, p_i donnera $(u_{ji}+v_{ji}, y_{u_{ji}}+y_{v_{ji}})$ aux joueurs lors de la phase $DevoileVal$. Les joueurs vérifieront ensuite si $(u_{ji}+v_{ji}, y_{u_{ji}}+y_{v_{ji}}, z_{u_{ji}}+z_{v_{ji}})$ est bien $1_\alpha-$*cohérant*.

Par contre, si au lieu du protocole de VI de CDDHR nous utilisions plutôt le protocole de VI de Rabin et Ben-Or, le joueur p_i qui calcule $u_{ji}+v_{ji}$ en privé devrait re-exécuter le PSV de CDDHR à partir de l'étape 3 pour obtenir les signatures-VI des sous-parts de ses nouvelles valeurs. Ceci est dû au fait que les vecteurs de vérification du protocole de VI de Rabin et Ben-Or ne sont pas linéaires.

Quoique non nécessaire pour la phase *Rec*, chaque joueur p_i doit aussi calculer les sommes $u_{ij} + v_{ij}$, $1 \leq j \leq N$, que les autres joueurs partagent avec lui. Ceci nous sera utile si nous utilisions plus tard un protocole de multiplication (et aussi nécessaire si nous utilisons un VI non homomorphe).

3.3 Multiplication par une constante

Nous voulons calculer $d \cdot u$ de façon multipartite où d est une constante publique et u est un secret mis en partage (de façon vérifiable) par la fonction f^u dont le terme constant est u. De façon semblable à l'addition, chaque joueur p_i calcule en privé $d \cdot u_{ji}$, $1 \leq j \leq N$, et se sert de la propriété de linéarité des signatures-VI de CDDHR pour rendre le secret $d \cdot u$ vérifiable.

$$\begin{pmatrix} d \cdot f^u(1,1) & \dots & d \cdot f^u(1,N) \\ & \vdots & \\ & d \cdot f^u(i,j) & \\ & \vdots & \\ d \cdot f^u(N,1) & \dots & d \cdot f^u(N,N) \end{pmatrix}$$

Si nous avions utilisé le VI de Rabin et Ben-Or, un receveur p_j aurait pu utiliser son vecteur de vérification (b, c) pour vérifier la signature-VI de p_i pour la valeur $d \cdot u_{ji}$ en vérifiant si $d \cdot u_{ji} + b_{ji}y = c$ (se référer à la section 2.5).

Comme dans le cas de l'addition, chaque joueur calcule aussi les valeurs $d \cdot u_{ij}$, $1 \leq j \leq N$, qui sont nécessaires si nous voulons utiliser les protocoles de multiplication par la suite.

3.4 Multiplication de deux secrets

Nous voulons calculer $u \cdot v$ où u et v sont mises en partage à l'aide d'un protocole PSV. Nous ne pouvons pas tout simplement demander à chaque participant p_i de calculer les produits $(u_{ji} \cdot v_{ji})$ de leurs parts de u et de v. Quoique ces points définissent bien le produit des polynômes des points (et donc un polynôme à terme constant $u \cdot v$), ce polynôme pourrait être de degré $2t > t+1$, violant ainsi le critère de l'interpolation du PSV. De plus, le polynôme n'est pas aléatoire, on n'obtient jamais un polynôme irréductible par exemple. Pour arriver au résultat voulu il nous faudra deux protocoles. Notons par $[s]_p^V$ un secret s mis en partage de façon vérifiable par un joueur p et $[s]^V$ un secret mis en partage de façon vérifiable par un joueur quelconque. Le premier protocole dont nous allons avoir besoin permet à un joueur p_i de prouver, pour trois secrets $[a]_{p_i}^V$, $[b]_{p_i}^V$ et $[c]_{p_i}^V$ que $c = ab$ sans dévoiler de l'information à propos de a, b ou c. Ainsi, chaque participant mettera en partage ses sous-parts u_{ji}, v_{ji} et $u_{ji} \cdot v_{ji}$, $1 \leq j \leq N$, de façon vérifiable et démontrera à l'aide du protocole que les produits sont bons. Le deuxième protocole que nous utiliserons permettra aux joueurs de transformer leurs parts résultantes de sorte à ce que $u \cdot v$ soit sous la forme d'un secret vérifiable mis en partage par un polynôme aléatoire de degré au plus t.

Vérification d'un produit

Nous décrivons un protocole VP-PSV utilisé par un initiateur D pour prouver que trois secrets $[a]_D^V$, $[b]_D^V$ et $[c]_D^V$ satisfont $ab = c$. Ce protocole a été utilisé pour la première fois dans le cadre des PSV dans [SS98], il avait déja apparu dans [CEvdG87] et [BCDP91] dans le cadre d'engagement basé sur le problème du logarithme discret. Il s'avère ainsi que le protocole est bon pour n'importe quel schéma d'engagement homomorphique (i.e. qui permet l'addition de secrets).

Protocole[VP-PSV]

VP-PSV$(D, P, [a]_D^V, [b]_D^V, [c]_D^V)$:

1. Pour $j = 1, \ldots k(t+1)$,
   (a) D choisit $b' \in_R K$ et calcule $c' = a \cdot b'$.
   (b) D se commet à b' et c' en calculant
      – $[b']_D^V \leftarrow PSV(D, b')$.
      – $[c']_D^V \leftarrow PSV(D, c')$.
   (c) Le participant $p_{j \ mod \ N}$ choisit $pile \in_R \{0, 1\}$:
      – Si $pile = 0$:
         – Les participants ouvrent $[b']_D^V$ (ils exécutent la phase *Rec* du PSV).
         – Ils calculent collectivement $[ab' - c']_D^V \leftarrow b' \cdot [a]_D^V - [c']_D^V$ (un calcul linéaire).
         – Ils ouvrent $[ab' - c']_D^V$ et vérifient que la valeur correspond bien à 0.
      – Si $pile = 1$:
         – Les participants calculent collectivement et ouvrent $[b+b']_D^V$.
         – Ils calculent $[a(b+b')-(c+c')]_D^V \leftarrow (b+b') \cdot [a]_D^V - [c']_D^V - [c]_D^V$ (calcul linéaire).
         – Ils ouvrent l'engagement et vérifient si la valeur égale 0.
   (d) Si une vérification échoue, terminer le protocole avec un **échec**.
2. Si aucun échec, terminer le protocole avec **succès**.

♠

Théorème 3.4.1 *Le protocole VP-PSV est une preuve à divulgation nulle qui permet à un participant p de démontrer que* $c = ab$ *à partir de* $[a]_p^V$, $[b]_p^V$ *et* $[c]_p^V$, *sans dévoiler de l'information à propos de a, b ou c. Le protocole est toujours exécuté avec succès lorsqu'en effet* $c = ab$ *et échoue avec probabilité au moins* $1 - 2^{-k}$ *si* $c \neq ab$.

Preuve: Divulgation nulle : Nous ne donons qu'une esquisse de preuve que les participants n'apprennent pas plus d'information à propos de a, b et c que ce qu'il pourrait déduire en connaissant que le résultat de la preuve à divulgation nulle. Nous supposons bien sûr que nous ne pouvons pas déduire de l'information de x à partir d'un engagement $[x]^V$. Notons qu'à chaque itération, soit $ab' - c'$ soit $a(b+b') - (c+c')$ (mais jamais les deux valeurs en même temps) est dévoilée. Dans les deux cas, si b' et c' sont choisis de façon uniformément aléatoire, on ne peut rien apprendre à propos de a, b et c à part le fait que c est équivalent à ab ou non. Dans les deux cas, les valeurs sont dissimulées par une fonction affine aléatoire. Notons que si les deux valeurs étaient dévoilées, il serait trivial de calculer $ab - c$, ce qui donnerait de l'information supplémentaire dans le cas ou $ab \neq c$. En effet, nous pouvons dévoiler le fait que $ab - c \neq 0$ (si c'est le cas, c'est ce qu'on veut prouver) mais il ne faut pas dévoiler la différence exacte !

Exactitude : Il est clair que si $c = ab$, la vérification à l'étape (c) sera toujours vraie et le protocole terminera avec succès.

Si par contre $c \neq ab$, il y a deux cas. Le premier, si D avait en effet choisi $c' = ab'$, les participants trouveront que $c + c' \neq ab + ab'$ et donc si $pile = 1$ le protocole échoue. Le deuxième, si $c \neq ab$ mais que $c' \neq ab'$, la vérification du cas $pile = 0$ échoue trivialement. Donc à chaque itération, si $pile$ est choisie de façon uniformément aléatoire et que $c \neq ab$, il y a une probabilité de $\frac{1}{2}$ que la vérification échouera. Après $(t + 1)k$ itérations, il y a eu au moins k tirages de $piles$ qui ont été faits de façon uniformément aléatoire (dans un groupe de $t + 1$ participants il y en a au moins 1 qui est honnête) et donc si $c \neq ab$ le protocole échoue avec probabilité $1 - 1/2^k$. $\square$

Réduction du degré du polynôme

Théorème 3.4.2 *Soit $p(x,y)$ un polynôme bivarié de degré $2t$ qui a pour terme constant $p(0,0) = c$. Supposons que chaque valeur $p(j,i)$, $1 \leq i,j \leq N$ est mise en partage par le joueur p_i à l'aide d'un PSV linéaire $(t+1, 2t)-$robuste. Il existe alors un protocole $(t+1, 2t)-$robuste pour mettre en partage le secret $c = p(0,0)$ utilisant un polynôme bivarié aléatoire de degré t.*

Preuve: Nous montrons comment construire ce protocole.
L'idée est que les participants généreront un polynôme aléatoire de degré $2t$ avec terme constant 0 dont les points seront distribués parmis les participants. Ils additionneront ensuite collectivement ce polynôme au polynôme dissimulant c et ensuite tronqueront pour obtenir un polynôme de degré au plus t (qui sera aléatoire). Tout ceci ce fait en n'effectuant que des opérations linéaires. Ce dernier polynôme sera celui qui mettera en partage la valeur c de façon vérifiable.
Génération d'un polynôme aléatoire : Pour générer un polynôme bivarié aléatoire $r(x,y)$ de degré au plus $2t$ en chacune de ces variables, tel que $r(0,0) = 0$, chaque joueur p_i distribue t^2 polynômes bivariés $g_{i,k,k'}(x,y)$ aléatoires (incluant le terme constant), $k, k' = 1, \ldots, t$, de degré au plus t en chaque variable.
Nous définissons

$$f_i(x,y) = \sum_{k=1}^{t} \sum_{k'=1}^{t} x^k y^{k'} \cdot g_{i,k,k'}$$

et laissons chaque joueur évaluer leurs points sur $f_i(x,y)$ à partir de leurs points de $g_{i,k,k'}(x,y)$.
Il est clair que les vecteurs des coefficients des monômes de $f_i(x,y)$ sont uniformément distribués et indépendants de l'information de n'importe quel ensemble d'au plus t joueurs. Il est aussi clair que $f_i(0,0) = 0$.
Ensuite, les joueurs calculent collectivement

$$r(x,y) = \sum_{i=1}^{N} f_i(x,y)$$

à l'aide du protocole d'addition multipartite vu à la section 3.2.

Notons ici qu'il serait impossible de construire collectivement un polynôme aléatoire de degré au plus t dont le terme constant est 0 et dont les monômes sont indépendants de l'information que possède au plus t joueurs. Pour constater ceci, il suffit de remarquer que tous les participants savent déja que $r(0,0) = 0$, ce qui leur donne à chacun un point sur la fonction de plus. Il suffirait donc que t participants se regroupent pour pouvoir interpoler le polynôme à partir des $t+1$ points qu'ils connaissent.

Maintenant, une fois que les participants ont généré les parts vérifiables (ainsi que les signatures-VI requises) du polynôme aléatoire $r(x, y)$, les participants calculeront collectivement la somme

$$p(x, y) + r(x, y).$$

Notons cette somme $q(x, y) := p(x, y) + r(x, y)$. Il est clair que le terme constant de q est c et que ses monômes sont uniformément aléatoires. Il suffit donc de tronquer ce polynôme.

Tronquer le polynôme : Pour ce faire, nous avons d'abord besoin de la formule d'interpolation de Lagrange pour les polynômes bivariés qui est donnée par

$$q(x, y) = \sum_{i=1}^{N} \sum_{i=j}^{N} p(i, j) L_{ij}(x)$$

où

$$L_{ij}(x) = \Pi_{s \neq i} \frac{x - s}{i - s} \Pi_{t \neq j} \frac{y - t}{j - t}.$$

Nous vérifions cette formule en remplaçant x par a et y par b, ce qui annule tous les termes de la somme sauf pour $i = a, j = b$, qui vaut $q(a, b)$. En sachant que le polynôme est unique, nous avons bien une expression pour ce polynôme.

Nous allons noter $\overline{q}$ le polynôme tronqué $q(x, y) \; mod \; x^{t+1} \; mod \; y^{t+1}$.

Nous pouvons écrire $\overline{q}$ comme une combinaison linéaire

$$\overline{q}(x, y) = \sum_{i=1}^{N} \sum_{j=1}^{N} p(i, j) \overline{L_{ij}}(x, y).$$

Notons que le polynôme $L_{ij}(x, y)$ (et ainsi $\overline{L_{ij}}(x, y)$) est connu publiquement puisqu'il ne dépend que des points fixes d'interpolation (que nous avons choisis comme étant $(1, 1), (1, 2), \ldots (N-1, N), (N, N)$.

Donc, pour fournir à chaque joueur i les valeurs $\overline{q}(i, 1), \ldots, \overline{q}(i, N)$ ainsi que $\overline{q}(1, i), \ldots, \overline{q}(N, i)$ et les signatures-VI appropriées, il suffit que les participants calculent collectivement $\overline{q}(x, y)$. Ceci n'est qu'un calcul linéaire sur les parts de $q(x, y)$ (déjà mises en partage) et se fait donc facilement à l'aide des protocoles de la section 3.2 et 3.3. $\square$

Partage des sous-parts

Une dernière chose qui nous manque est une façon de permettre à un joueur de mettre en partage ces sous-parts $u_{i1}, \ldots, u_{iN}$ de sorte à ce que les autres participants soient assurés qu'au moins $t+1$ de ces sous-parts aient vraiment été mises en partage (et pas d'autres valeurs incohérentes). Imaginons que p_i partage des valeurs a et b qu'il choisit à son aise, le fait de prouver qu'en effet $a \cdot b = c$ ne nous avance en rien.

Rappelons-nous que chaque valeur u_{ji} de p_i est aussi connue par le joueur p_j et que p_i possède une signature-VI de ce dernier pour cette valeur. Nous utilisons ce fait pour construire l'algorithme qui suit :

Protocole[PSV-sous-parts]

PSV-sous-parts(P, PSV-CDDHR(s)) :

1. Chaque participant p_i partage à l'aide du PSV de CDDHR chacun de ses sous-parts $s_{1i}, \ldots s_{Ni}$. Notons le résultat par les engagements $[s_{1i}]^V_{p_i}, \ldots [s_{Ni}]^V_{p_i}$.
   Chaque participant p_i partage aussi ses valeurs de vérification $s_{i1}, \ldots, s_{iN}$.
2. Pour chaque participant p_i, Pour chaque participant p_j,
   - Calculer collectivement $[s_{ji}]^V_{p_i} - [s_{ji}]^V_{p_j}$.
   - Dévoiler le résultat et vérifier qu'il est bien 0.

– Si le résultat n'est pas 0, p_i doit dévoiler sa valeur $s_{ij}^{'}$ ainsi que la signature-VI de p_j qu'il possède pour cette valeur.
Les participants dévoilent ensuite la valeur de $[s_{ij}]_{p_i}^{V}$. Si $s_{ij}^{'} \neq s_{ij}$ et que la signature-VI de p_j pour $s_{ij}^{'}$ est invalide (ou que p_i décide de ne pas collaborer) p_i est disqualifié.

♠

Théorème 3.4.3 *À la fin du protocole $PSV - sous - parts$, chaque participant aura mis en partage au moins $t+1$ valeurs correspondant vraiment à leurs sous-parts de s, sans dévoiler d'information à propos de s à l'adversaire.*

Preuve:

Exactitude Il est clair que si un joueur p_i essaie de s'engager à une valeur $\overline{s_{ji}}$ qui est différente de sa part réelle (s_{ji}) qu'il partage avec un joueur honnête p_j , p_j se plaindra à l'étape de vérification. Les participants dévoileront alors la valeur $\overline{s_{ji}}$ et p_i publiera une signature-VI (venant supposément de p_j) qui n'est pas pour cette valeur.
Une signature-VI de p_j ne peut pas être fabriquée par p_i (si p_j est honnête) avec probabilité de succès plus grande que $O(2^{-k+lg\,n})$. Donc si p_j est honnête, avec probabilité $\geq 1 - O(2^{-k+lg\,n})$ p_i se fera disqualifier.
Puisqu'il y a au moins $t + 1$ joueurs honnêtes, p_i devra mettre en partage au moins $t + 1$ véritables sous-parts pour ne pas se faire disqualifier.
Confidentialité : Notons simplement que si une part $\overline{s_{ji}}$ est dévoilée, soit p_i soit p_j est malhonnête (un des deux s'est engagé à une mauvaise valeur. L'adversaire n'apprend donc aucune information supplémentaire puisque seule la valeur $\overline{s_{ji}}$ est dévoilée et que celle-ci était déjà connue par l'adversaire. □

Protocole de multiplication

Nous pouvons finalement décrire le protocole de multiplication.

Protocole[MULT]

Distr$(P, [u]^V, [v]^V)$:

- Chaque joueur p_i partage ses sous-parts $u_{1i}, \ldots u_{Ni}$ de u ainsi que ses sous-parts $v_{1i}, \ldots v_{Ni}$ de v à l'aide du protocole PSV-sous-parts.
- Chaque joueur p_i calcule $w_{ji} \leftarrow u_{ji} \cdot v_{ji}$, $1 \leq j \leq N$, en privé et partage ses valeurs à l'aide du PSV de CDDHR.
- Chaque joueur p_i prouve à l'aide du protocole VP-PSV que les engagements $[w_{ji}]^V_{p_i}$, $1 \leq j \leq N$ satisfont bien $w_{ji} = u_{ji} \cdot v_{ji}$.
- Exécuter le protocole du théorème de la section 3.4.2.

♠

Le seul problème qui pourrait survenir est à l'étape 1, où un joueur pourrait décider de ne pas mettre en partage ses sous-parts. Si cela arrive, les joueurs ouvrent collectivement les sous-parts du joueur malhonnête et le protocole recommence à l'étape du partage des sous-parts. Cela prolongera le temps d'exécution du protocole par au plus un facteur de N.

Remarquons que le protocole en surface n'a aucune probabilité d'erreurs. Sa probabilité d'erreur est au plus la somme des probabilité d'erreur de ses sous-protocoles. Puisque chaque protocole a une probabilité d'erreur exponentiellement petite en k et qu'ils sont exécutés un nombre polynomial de fois, la probabilité d'erreur reste exponentiellement petite en k.

Chapitre 4

Adversaire généralisé

Jusqu'ici, nous n'avons traité que des adversaires à seuil, c'est à dire d'adversaires qui contrôlent un nombre de participants ne dépassant pas une certaine limite. Par exemple, nous avons donné au chapitre 3 un protocole PCM qui tolère un adversaire à seuil contrôlant moins de $n/2$ membres. Cependant, d'autres structures plus générales d'adversaires peuvent être tolérées par un PCM. Imaginons un scénario où nous avons 6 participants, $P = \{p_1, p_2, ..., p_6\}$ qui veulent calculer une fonction de façon privée (i.e. respectant les critères d'un PCM sûr). Il est possible que certains joueurs, si regroupés, veuillent collaborer entre eux pour acquérir de l'information, de façon malhonnête, à propos des données privées des autres joueurs, tandis que les participants d'autres coalitions seraient honnêtes. Disons, par exemple, que les joueurs des ensembles $\{p_1, p_2, p_3, p_4\}$, $\{p_1, p_2, p_4\}$, $\{p_1, p_3, p_5\}$, $\{p_2, p_5\}$, $\{p_4, p_6\}$ se regroupent pour tenter d'obtenir de l'information à propos des données privées des autres joueurs. Est-il possible pour les 6 joueurs de calculer une fonction de façon privée ? Hirt et Maurer ([HM97]) ont montré que c'est en effet possible ! Il suffit d'affecter un poids de travail w_i (un nombre entier) à chaque participant p_i et de lui assigner la tâche de w_i participants dans un protocole standard avec plus de joueurs. De cette façon, il est possible de procéder à un calcul multipartite sécuritaire contre un adversaire actif si et seulement si

$$\sum_{i:\, p_i \in P \text{ est malhonnête}} w_i < \frac{1}{2} \sum_{i:\, p_i \in P} w_i.$$

Dans notre exemple, p_1, p_2, p_3 et p_4 auraient le poids de travail $w_1 = w_2 = w_3 = w_4 = 1$, p_5 se verrait attribuer $w_5 = 2$ et finalement p_6 aurait le poids $w_6 = 3$.
Il est en effet possible de calculer une fonction de façon privée si et seulement si aucune paire d'ensembles de coalitions malhonnêtes ne somment (avec l'opérateur d'union d'ensemble) à l'ensemble total des joueurs P. Nous notons un adversaire obéissant à une telle contrainte par $\mathcal{Q}^2$. Cette notion se généralise à la définition suivante :

Définition: 4.0.1 *Une structure d'adversaire $\mathcal{A}$ sur P, $\mathcal{A} \subseteq 2^P$, est dite $\mathcal{Q}^k$ si aucuns k ensembles de A ne somment à P, c'est-à-dire que*

$$\nexists_{A_1,\ldots,A_k \in A}\ A_1 \cup \ldots \cup A_k = P.$$

♣

Notons que $\mathcal{Q}^k \Rightarrow \mathcal{Q}^{k+1}$.
Hirt et Maurer ont en fait étendu les résultats de [BGW88], [CCD88] et [RB89] (voir le chapitre 1), qui étaient sûrs contre des adversaires à seuil $n/3$ et $n/2$, à des PCM sûrs contre des adversaires de structure $\mathcal{Q}^3$ et $\mathcal{Q}^2$ respectivement. Par la suite, se basant sur les *span programs* (nous les décrirons plus loin), Cramer, Damgård et Maurer ([CDM98]) ont donné un PCM inconditionnellement sûr contre un adversaire de structure $\mathcal{Q}^3$ qui est super-polynomialement plus efficace en N que celui de [HM97] pour certains types de structures d'adversaire. Plus précisément, le PCM de [HM97] a un temps d'exécution polynomial en $|A|$, qui est souvent polynomial en N, tandis que le PCM de [CDM98] est polynomial en N. Ces derniers auteurs ont aussi exhibé un premier protocole PCM cryptographiquement sûr contre un adversaire de type $\mathcal{Q}^2$. Beaver et Wool [BW98] ont aussi amélioré les résultats de [HM97] par un facteur polynomial, mais seulement dans le cas d'un adversaire passif. Par la suite, Smith et Stiglic ([SS98]) ont présenté un protocole PCM inconditionnellement sûr contre un adversaire actif

$\mathcal{Q}^2$, se basant sur les *span programs*, qui est aussi super-polynomialement plus efficace que celui du même type présenté dans [HM97]. Ces derniers ont utilisé le schéma de [RB89], modifiant le protocole de vérification de multiplication de parts partagées de façon vérifiable pour qu'il soit efficace. L'efficacité du protocole de [RB89] est polynomiale en $|K|^2$, tandis que celui de [SS98] est polynomiale en $log|K|$. Finalement, le PCM introduit dans [CDD$^+$99] est aussi généralisable à des adversaires de structure $\mathcal{Q}^2$. Cette généralisation donne un PCM encore plus efficace que celui de [SS98] (la complexité du premier est dans $O((k+log\ n)nm^3)$ tandis que celui du dernier est dans $O(k^2(k+log\ n)nm^3)$.

Nous présentons dans ce chapitre les *span programs* ainsi que la construction du schéma de partage de secret induit par ce dernier. Finalement, nous présentons l'esquisse de la construction d'un PCM, efficace, inconditionnellement sûr contre des adversaires de structure $\mathcal{Q}^2$.

4.0.1 *Span programs*

Un *span program* est un modèle calculatoire basé sur l'algèbre linéaire qui a été présenté pour la première fois par Karchmer et Wigderson dans [KW93]. Un *span program* calcule de façon naturelle une fonction booléenne. Le modèle se définit en fait par une matrice M, sur un corps K, étiquetée par une certaine fonction ρ qui associe à chaque ligne M_i de M un littéral x_i^ϵ où $x_i^1 = x_i$ et $x^0 = \overline{x_i}$. Formellement, nous avons la définition suivante :

Définition: 4.0.2 (*Span program) Soit un corps K. Un span program est un triplet $\hat{M} := (M, \rho, K)$ où M est une matrice $d \times e$ sur le corps K et ρ est une fonction des indices de ligne de M vers l'ensemble des littéraux $\{x_i^\epsilon | i \in \{1, \ldots, n\}, \epsilon \in \{0, 1\}\ \}$ qui étiquette les lignes de M.* ♣

Notons que la fonction ρ n'associe pas nécessairement une ligne i à un littéral d'indice i.

La taille de $\hat{M}$ est définie comme le nombre de colonnes de M.

On définit le *span* de M ($span(M)$) comme le sous-espace engendré par les lignes de M. De même, M_σ, pour $\sigma \in \{0,1\}^n$, représente la matrice formée des lignes i de M telles que l'évaluation de x_i^ϵ au point σ_i soit vérifiée (donne la valeur logique vraie). Nous dirons que le *span program* accepte une entrée σ si et seulement si $\vec{v}$ est dans le *span* de M_σ, où $\vec{v}$ est un vecteur fixe, choisi de façon quelconque selon un changement de base approprié . Nous prendrons le vecteur $\vec{1} := (1, 0, \ldots, 0)$. En conséquence, chaque *span program* calcule une fonction booléenne

$$f(\sigma) = \begin{cases} 1 \ si \ \vec{1} \in span(M_\sigma), \\ 0 \ sinon. \end{cases}$$

Une fonction $f : A \rightarrow B$ est dite monotone si et seulement si pour tous $x, y \in A$ nous avons

$$x \leq_A y \Rightarrow f(x) \leq_B f(y)$$

où $\leq_A$ et $\leq_B$ sont des relations d'ordre partiel sur A et B respectivement.
Nous pouvons définir un ordre partiel sur $\{0,1\}^n$ par la règle : $x \leq y$ si et seulement si chaque coordonnée de x est plus petite que la coordonnée correspondante de y.
Dans la même veine, nous avons aussi une notion de monotonie pour les *span programs* :

Définition: 4.0.3 *Un span program $\hat{M} := (M, \rho, K)$ est dit monotone si l'image de ρ ne se compose que des littéraux positifs $\{x_1, \ldots, x_n\}$.* ♣

Il est évident qu'un *span program* monotone calcule une fonction booléenne monotone. Cependant, nous pouvons aussi démontrer que toute fonction monotone peut être calculée par un *span program* monotone.

Exemple: 4.0.4 *Le span program*

$$(\begin{pmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}, \rho, GF(2))$$

où $\rho(1) = x_1$ et $\rho(2) = x_2$, calcule le prédicat logique $\wedge$.

Si nous associons l'ensemble décrit par l'expression régulière $\{0,1\}^n$ avec l'ensemble des sous-ensembles $\wp(\{1,\ldots,n\})$, la relation $\leq$ correspond à la relation d'inclusion $\subseteq$, dans $\wp(\{1,\ldots,n\})$, de façon à ce que pour deux ensembles $A, B \in \wp(\{1,\ldots,n\})$ nous avons $A \subseteq B \Rightarrow f(A) \leq f(B)$.
Pour cette raison, une fonction monotone f définit aussi de façon naturelle une structure d'adversaire $\mathcal{A}_f = \{B \subseteq P | f(B) = 0\}$.
De plus, étant donné un *span program* calculant f, il existe un schéma de partage de secret linéaire qui tolère la structure d'adversaire $\mathcal{A}_f$. Nous appelons alors $\Gamma_f = \wp(P) - \mathcal{A}_f$ la structure d'accès du protocole de PS, c'est en fait l'ensemble des ensembles de participants qui peuvent reconstruire le secret. D'ailleurs, nous avons $\Gamma_f = \{B \subseteq P | f(B) = 1\}$. En vue de décrire le schéma de PS, nous allons définir une fonction surjective qui associe à chaque ligne d'une matrice un participant de P. En effet, nous pouvons observer que la fonction d'étiquetage d'un *span program* monotone ne fait qu'associer à une ligne de la matrice un nombre dans $\{1,\ldots,n\}$. Nous pouvons donc définir la fonction $\breve{\rho} : \{1,\ldots,n\} \rightarrow P$ suivante :

$$\breve{\rho}(i) = p_j \iff \rho(i) = x_j^{\epsilon},\ \epsilon \in \{0,1\}.$$

C'est-à-dire, $\breve{\rho}(i)$ associe à la ligne i de la matrice M d'un *span program* (M, ρ, K) un joueur p_j si σ étiquette le litéral x_j^{ϵ} à la ligne i de M. La fonction peut associer à plusieurs lignes un même joueur et tout joueur a au moins une ligne qui lui est associée (la fonction est surjective mais pas nécessairement bijective). Ceci impose bien sûr que le nombre de ligne de la matrice, d, doit être $\geq N$.

Nous pouvons maintenant décrire le schéma de partage de secret linéaire sûr contre un adversaire de structure $\mathcal{A}$ défini par un *span program* monotone $\hat{M}$: (M, ρ, K), où M est une matrice $d \times e$:

Protocole[Partage de Secret généralisé]

Distr$(D, P, s, \hat{M})$:

- D choisit $e-1$ valeurs $\rho_1, \ldots, \rho_{e-1} \in_R K$ pour former le vecteur ligne $\vec{b} = (s, \rho_1, \ldots \rho_{e-1})$.

- D calcule le vecteur $\alpha \leftarrow M\vec{b}^T$.
- D distribue $\alpha_i \leftarrow M_i\vec{b}^T$ au joueur $p_{\tilde{\rho}}(i)$, pour $1 \leq i \leq d$.

♠

Théorème 4.0.4 *Le protocole ci-haut est un PS sûr contre un adversaire de structure $\mathcal{A}$ défini par le span program $\hat{M}$ utilisé.*

Preuve: Soit f la fonction monotone calculée par $\hat{M}$. Soit $\Gamma_f = \wp(P) - \mathcal{A}_f$ où $\mathcal{A}_f$ est la structure d'adversaire définie par $\hat{M}$.

Nous montrons d'abord qu'un ensemble de participants $B \subseteq P$ peut reconstruire le secret s si et seulement si leur colonne de M (M_B) contient dans leur *span* le vecteur $\vec{1} = (1, 0, \ldots, 0)$ (donc si et seulement si l'ensemble fait partie de Γ_f). En effet, supposons qu'il existe un vecteur λ tel que $\lambda M_A = \vec{1}^T$, alors

$$\langle \lambda, \alpha_A \rangle = \lambda M_A \vec{b}^T = \vec{1}\,\vec{b}^T = s,$$

où $\langle , \rangle$ est le produit interne standard sur K^d.

Notons que λ est calculable efficacement.

En dernier lieu, nous devons démontrer qu'un ensemble A de participants dont les colonnes ne font pas partie du *span* de $\hat{M}$ (c'est-à-dire un ensemble qui est un élément de la structure d'adversaire $\mathcal{A}_f$) ne peut acquérir d'information à propos du secret s.

Notons d'abord que l'image de M^T est égale au complément orthogonal du noyau de M. En particulier, cela veut dire qu'il existe un vecteur κ telle que $M_A\kappa^T = \vec{0}$ et $\kappa_1 = \vec{1}$. Alors, pour toute valeur $\vec{q} \in K^d$, telle que $\vec{q}^T = M_A\vec{w}^T$ pour n'importe quel $\vec{w} \in K^e$, nous avons aussi que $M_A(\vec{w}(j))^T = \vec{q}$, où $w(j) = \vec{w} + j\kappa$, pour tout $j \in K$. Mais les valeurs de $\vec{1}(\vec{w}(j))^T$ sont toutes distinctes, engendrant tous les éléments de K, ce qui indique que l'ensemble A n'a aucune information à propos de s. □

4.0.2 Calculs multipartites contre adversaire généralisé

En remplaçant, dans le PSV du chapitre 3, le PS de Shamir par le protocole de PS généralisé, nous obtenons un protocole légèrement plus faible que le protocole de PSV que nous avons introduit. En effet, le protocole de PSV présenté ne partageait pas seulement une valeur secrète, mais bien une fonction dont le terme constant est le secret. D'ailleurs, Gennaro, Rabin et Rabin ont attribué le nom de "partage de secret et de polynôme vérifiable" (PSPV) à ce genre de schéma. A priori, un nouveau modèle de partage de secret est introduit dans leur article [GRR98], tous les protocoles de partage de secret partagent aussi un polynôme. Le polynôme ainsi partagé par notre PSV est en fait utilisé dans les schémas de calcul de fonctions. Nous avons besoin de partager ce polynôme puisque c'est par des calculs sur les parts du polynôme partagé que les participants arrivent à calculer le résultat voulu. Nous avons aussi besoin de reconstituer ce polynôme dans le cas, par exemple, où un participant sous le contrôle d'un adversaire décide de ne pas collaborer en refusant de suivre les étapes du protocole. Bref, si nous remplaçons le protocole de PS de Shamir par le nouveau, un ensemble de participants de la structure d'accès pourrait reconstruire la valeur du secret, mais pas le vecteur par lequel il a été partagé. Heureusement, des protocoles de calculs linéaires qui ont été introduits dans [CDM98] nous portent secours. En effet, à l'aide d'un protocole de transfert d'engagement (PTE), qui transfert l'engagement d'un joueur p_i à une valeur a ($[a]_{p_i}$) à un autre joueur p_j, résultant à l'engagement $[a]_{p_j}$, ainsi que d'un protocole de partage d'une part (PPP) d'un engagement, nous pouvons construire le protocole PSV qui a les propriétés voulues. Cela a d'ailleurs été suggéré dans [CDD+99].

Cela dit, le protocole PSV généralisé que nous pouvons construire permet de calculer n'importe quelle fonction linéaire de façon sécuritaire à l'aide des protocoles d'addition et de multiplication multipartites présentés au chapitre 3. La multiplication, par contre, est encore une fois une tâche non triviale. Il semble bien que l'implantation du processus de multiplication multipartite ne soit pas un phénomène linéaire sans recours à la communication. Cependant, il existe une propriété des *span programs* (caractérisée pour la première fois dans [CDM99]) qui admet la multiplication. Cette propriété peut d'ailleurs être acquise par n'importe quelle structure d'accès $\mathcal{Q}^2$.

Définition: 4.0.5 (*Span program* **avec multiplication)** *Un span program* $\hat{M}$: (M, ρ, K) *possède la propriété de multiplication s'il existe un vecteur* $\vec{r}$ *(dénommé un vecteur de recombinaison) tel que*

$$\forall_{\vec{b},\vec{b'} \in K^e} \langle \vec{r}, M\vec{b} * M\vec{b'} \rangle = \langle \vec{1}, \vec{b} * \vec{b'} \rangle$$

où $\langle \cdot, \cdot \rangle$ *est le produit interne standard sur* K^e *et pour* $\vec{v} = (v_1, \cdots, v_d)$ *et* $\vec{w} = (w_1, \cdots w_d)$ *nous avons* $v * w = (v_1 w_1, \ldots, v_d w_d)$. ♣

Ainsi, l'idée de la construction du protocole de multiplication multipartite de deux secrets partagés a, b est de demander encore une fois à chaque participant $p_{\breve{\rho}(i)}$ de mettre en partage chacune de ses sous-parts $a_{i_1}, \ldots, a_{i_d}$ et $b_{i_1}, \ldots, b_{i_d}$. Ensuite, le participant calcule $a_{i_j} \cdot b_{i_j}$, pour $1 \leq j \leq d$, et prouve (à l'aide du protocole de la section 3.4 par exemple) que le produit est correct. Il s'engage alors à ses produits à l'aide du PSV. Pour reconstituer le secret, les participants calculent simplement, de façon collective,

$$[ab]^V = r_1 * [c_1]^V_{\breve{\rho}(1)} + \ldots + r_d * [c_d]^V_{\breve{\rho}(d)},$$

où $\vec{r} = (r_1, \ldots, r_d)$ est le vecteur de recombinaison qui est connu publiquement. Comme pour notre multiplication multipartite du chapitre 3, si un participant refuse de mettre en partage ses parts les autres joueurs les reconstruisent et nous reprenons le protocole à partir du début. Cette attaque ne prolonge le protocole que par un facteur au plus N. De plus, un résultat récent de Cramer, Damgård et Maurer ([CDM99]) démontre que pour n'importe quelle structure d'accès $\mathcal{Q}^2$, la complexité du *span program* la définissant équivaut à la complexité du *span program* avec multiplication qui lui est associé. Ce dernier résultat nous assure que la construction esquissée ci-haut donne un PCM efficace (sa complexité est dans l'ordre de $max(Poly(k), m^4)$, où m est la taille du *span program* monotone).

Chapitre 5

Conclusion

Les chapitres précédents ont présenté et développé les notions de calculs multipartites dans un modèle inconditionnellement sûr avec canal de diffusion publique. Ce modèle a été choisi car il permet une meilleure abstraction du problème et permet ainsi d'obtenir des résultats très généraux. Nous avons discuté en détail de divers sous-protocoles qui ont menés au développement des schémas de calculs multipartites inconditionellements sûrs les plus efficaces connue à ce jour. Nous avons aussi décrit un nouveau protocole de calucul multipartite inconditionnellement sûr, très intuitif, qui met en valeur les notions les plus importantes de ce type de calcul. Ce dernier protocole peut être adapté pour assurer la sécurité contre un adversaire plus général, tel que présenté au dernier chapitre. Tout au long de ce mémoire, nous avons utilisé un formalisme qui est souvent manquant dans la littérature dans ce domaine, la sécurité de tous les protocoles décrits a été prouvée.

Il reste cependant encore plusieurs sujets à développer en calcul multipartite. Nous pensons aux techniques utilisées dans des schèmes résolvant des problèmes plus précis, comme par exemple les votes. Dans un scrutin, on doit s'assurer que seules les personnes autorisées aient le droit de voter. On ne doit pas permettre à une personne de voter plus d'une fois ou de vendre son vote et une personne doit pouvoir s'assurer que son vote soit compté. Le critère de protection contre la fraude est le plus difficile à respecter et aucune technique que nous connaissons ne le fait de façon adéquate. Nous pensons également à l'évaluation d'autres fonctions spécifiques, les fonctions de chiffrements, de signatures numériques. Nous devons développer des techniques précises dans le but d'optimiser ces calculs. Nous pourrons peut-être modéliser d'autres types d'adversaires aussi, et fournir des fonctionnalités qui sont sûres contre une attaque dirigée par ces genres d'adversaires.

Nous croyons que le domaine des calculs multipartites est grand ouvert, autant du côté du développement pratique que théorique. Nous espérons surtout que les techniques procurant la confidentialité des données privées continueront toujours de progresser dans une voie formelle, avec preuve de leur sécurité.

Bibliographie

[BCDP91] J. Boyar, D. Chaum, I. Damgård, and T. Pederson. Convertible undeniable signatures. In Alfred J. Menezes and Scott A. Vanstone, editors, *Advances in cryptology — CRYPTO '90 : proceedings*, volume 537 of *Lecture Notes in Computer Science*, pages 189–205, Berlin, Germany / Heidelberg, Germany / London, UK / etc., 1991. Springer-Verlag.

[Bea90] D. Beaver. Multiparty protocols tolerating half faully processors. In *Advances in Cryptology : CRYPTO '89*, pages 560–572, Berlin, August 1990. Springer.

[BGW88] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for noncryptographic fault-tolerant distributed computations. In ACM, editor, *Proceedings of the twentieth annual ACM Symposium on Theory of Computing, Chicago, Illinois, May 2–4, 1988*, pages 1–10, New York, NY 10036, USA, 1988. ACM Press.

[BH92] D. Beaver and S. Haber. Cryptographic protocols provably secure against dynamic adversaries. In R. A. Rueppel, editor, *Advances in Cryptology—EUROCRYPT 92*, volume 658 of *Lecture Notes in Computer Science*, pages 307–323. Springer-Verlag, 24–28 May 1992.

[Bla79] G. R. Blakley. Safeguarding cryptographic keys. In Richard E. Merwin, Jacqueline T. Zanca, and Merlin. Smith, editors, *1979 National Computer Conference : June 4–7, 1979, New York, New York*, volume 48 of *AFIPS Conference proceedings*, pages 313–317, Montvale, NJ, USA, 1979. AFIPS Press.

[BW] E. Berlekamp and L. Welch. Error correction of algebraic block codes. US Patent Number 4,633,470.

[BW98] D. Beaver and Wool. Quorum-based secure multi-party computation. In *EUROCRYPT : Advances in Cryptology : Proceedings of EUROCRYPT*, 1998.

[Can95] R. Canetti. *Studies in Secure Multiparty Computation and Applications.* PhD thesis, Department of Computer Science and Applied Mathematics, The Weizmann Institute of Science, June 1995. revised version.

[CCD88] D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols (extended abstract). In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, pages 11–19, Chicago, Illinois, 2–4 May 1988.

[CDD+99] R. Cramer, I. Damgård, S. Dziembowski, M. Hirt, and T. Rabin. Efficient multiparty computations with dishonest minority. In *Advances in Cryptology—EUROCRYPT 99*, volume 1561 of *Lecture Notes in Computer Science*, pages 311–326. Springer-Verlag, March 1999.

[CDM98] R. Cramer, I. Damgård, and U. Maurer. Span programs and general multiparty computation. Dernière version disponible des auteurs, 1998.

[CDM99] R. Cramer, I. Damgård, and U. Maurer. Communication privée. Communication privée, 1999.

[CDvdG87] D. Chaum, I. B. Damgård, and J. van de Graaf. Multiparty computations ensuring privacy of each party's input and correctness of the result. In Carl Pomerance, editor, *Advances in Cryptology—CRYPTO '87*, volume 293 of *Lecture Notes in Computer Science*, pages 87–119. Springer-Verlag, 1988, 16–20 August 1987.

[CEvdG87] D. Chaum, J.-H. Evertse, and J. van de Graaf. An improved protocol for demonstrating possession of discrete logarithms and some generalizations. In David Chaum and Wyn L. Price, editors, *Advances in Cryptology—EUROCRYPT 87*, volume 304 of *Lecture Notes in Computer Science*, pages 127–141. Springer-Verlag, 1988, 13–15 April 1987.

[CFGN96] R. Canetti, U. Feige, O. Goldreich, and M. Naor. Adaptively secure multi-party computation. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, pages 639–648, Philadelphia, Pennsylvania, 22–24 May 1996.

[CFSY96] R. Cramer, M. Franklin, B. Schoenmakes, and M. Yung. Multi-authority secret-ballot elections with linear work. In Ueli Maurer, editor, *Advances in Cryptology—EUROCRYPT 96*, volume 1070 of *Lecture Notes in Computer Science*, pages 72–83. Springer-Verlag, 12–16 May 1996.

[CGKS95] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *36th Annual Symposium on Foundations of Computer Science (FOCS'95)*, pages 41–51, Los Alamitos, October 1995. IEEE Computer Society Press.

[CGMA85] B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults. In IEEE, editor, *26th annual Symposium on Foundations of Computer Science, October 21–23, 1985, Portland, OR*, pages 383–395, 1109 Spring Street, Suite 300, Silver Spring, MD 20910, USA, 1985. IEEE Computer Society Press.

[Cha90] D. Chaum. The spymasters double-agent problem : Multiparty computations secure unconditionally from minorities and cryptographically from majorities. In *Advances in Cryptology : CRYPTO '89*, pages 591–603, Berlin, August 1990. Springer.

[DDFY94] A. De Santis, Y. Desmedt, Y. Frankel, and M. Yung. How to share a function securely. In ACM, editor, *Proceedings of the twenty-sixth annual ACM Symposium on the Theory of Computing : Montréal, Québec, Canada, May 23–25, 1994*, pages 522–533, New York, NY 10036, USA, 1994. ACM Press.

[FHM98] M. Fitzi, M. Hirt, and U. Maurer. Trading correctness for privacy in unconditional multi-party computation. *Lecture Notes in Computer Science*, 1462 :121–137, 1998.

[FHM99] M. Fitzi, M. Hirt, and U. Maurer. Trading correctness for privacy in unconditional multi-party computation : Corrected version. Available from authors, 1999.

[FM88] P. Feldman and S. Micali. Optimal algorithms for Byzantine agreement. In *Proc. 20th Ann. ACM Symp. on Theory of Computing*, pages 162–172, 1988. The expected running time of this algorithm is constant in a synchronous network of n nodes if the number of faults is less than $n/3$, and in an asynchronous network of n nodes if the number of faults is less than $n/4$.

[FOO93] A. Fujioka, T. Okamoto, and K. Ohta. A practical secret voting scheme for large scale elections. *Lecture Notes in Computer Science*, 718 :244–? ?, 1993.

[FR96] M. K. Franklin and M. K. Reiter. The design and implementation of a secure auction service. In *IEEE Transactions on Software Engineering*, volume 22(5), pages 302–312, May 1996.

[GJKR96a] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust and efficient sharing of RSA functions. In Neal Koblitz, editor, *Advances in cryptology, CRYPTO '96 : 16th annual international cryptology conference, Santa Barbara, California, USA, August 18–22, 1996 : proceedings*, volume 1109 of *Lecture Notes in Computer Science*, pages 157–172, Berlin, Germany / Heidelberg, Germany / London, UK / etc., 1996. Springer-Verlag.

[GJKR96b] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust threshold DSS signatures. In Ueli Maurer, editor, *Advances in cryptology, EUROCRYPT '96 : International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12–16, 1996 : proceedings*, volume 1070 of *Lecture Notes in Computer Science*, pages 354–371, Berlin, Germany / Heidelberg, Germany / London, UK / etc., 1996. Springer-Verlag.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game — A completeness theorem for protocols with honest majority. In ACM, editor, *Proceedings of the nineteenth annual ACM Symposium on Theory of Computing, New York City, May 25–27, 1987*, pages 218–229, New York, NY 10036, USA, 1987. ACM Press.

[GRR98] R. Gennaro, M. O. Rabin, and T. Rabin. Simplified VSS and fast-track multiparty computations with applications to threshold cryptography. In *Proc. 17th ACM Symposium on Principles of Distributed Computing (PODC)*, 1998.

[GS92] Peter Gemmell and Madhu Sudan. Highly resilient correctors for polynomials. *Information Processing Letters*, 43(4) :169–174, September 1992.

[HM97] M. Hirt and U. Maurer. Complete characterization of adversaries tolerable in general multiparty computations. In *Proc. ACM PODC'97*, pages 25–34, 1997.

[IK99] Y. Ishai and E. Kushilevitz. Improved upper bounds on information-theoretic private information retrieval (extended abstract). In *STOC : ACM Symposium on Theory of Computing (STOC)*, 1999.

[KO97] E. Kushilevitz and R. Ostrovsky. Replication is NOT needed : SINGLE database, computationally-private information retrieval. In

Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS-97), pages 364–373, Los Alamitos, October 20–22 1997. IEEE Computer Society Press.

[KW93] M. Karchmer and A. Wigderson. On span programs. In *Proc. of Structure in Complexity*, pages 102–111, 1993.

[MVV97] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone. *Handbook of applied cryptography.* The CRC Press series on discrete mathematics and its applications. CRC Press, 2000 N.W. Corporate Blvd., Boca Raton, FL 33431-9868, USA, 1997.

[Ray00] J.-F. Raymond. Private data-base queries. Master thesis, McGill University, 2000. to be published.

[RB89] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In ACM, editor, *Proceedings of the Eighth Annual ACM Symposium on Principles of Distributed Computing : Edmonton, Alberta, Canada, August 14–16, 1989*, pages 73–85, New York, NY 10036, USA, 1989. ACM Press.

[Sha48] C. E. Shannon. A mathematical theory of communication. volume 27, pages 379–423, 623–656, 1948.

[Sha79] A. Shamir. How to share a secret. *Communications of the Association for Computing Machinery*, 22(11) :612–613, November 1979.

[Spi94] M. Spivak. *Calculus (3rd edition).* W. A. Benjamin, Inc., New York, 1994.

[SS98] A. Smith and A. Stiglic. Multiparty computation unconditionally secure against Π^2 adversary structures. Cryptology SOCS-98.2, School of Computer Science, McGill University, Montreal, Canada, 1998.

[Sti95] D. R. Stinson. *Cryptography Theory and Practice.* CRC Press, Boca Raton, 1995.

[Tap95] A. Tapp. Evaluation de fonctions sur données privées. Mémoire de maîtrise, Université de Montréal, August 1995.

[Yao82] A. C. Yao. Protocols for secure computations (extended abstract). In *23th Annual Symposium on Foundations of Computer Science*, pages 160–164, Chicago, Illinois, November 1982. IEEE Computer Society Press.