Méthodes dissuasives contres les utilisateurs malhonnêtes dans les systèmes répartis

Kévin Huguenin

N° d'ordre : 4279 ANNÉE 2010

UNIVERSITÉ DE RENNES 1

THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Européenne de Bretagne

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Informatique

École doctorale Matisse

présentée par

Kévin HUGUENIN

préparée à l'unité de recherche IRISA – UMR 6074
Institut de Recherche en Informatique et Systèmes Aléatoires

Méthodes dissuasives contre les utilisateurs malhonnêtes dans les systèmes répartis

Thèse soutenue à Rennes
le 10 Décembre 2010

devant le jury composé de :

Luc BOUGÉ
Professeur des universités, ENS Cachan / *Président*
Carole DELPORTE-GALLET
Professeure des universités, Université Paris 7 / *Rapporteuse*
Pablo RODRIGUEZ
Scientific director, Telefonica I&D / *Rapporteur*
Rachid GUERRAOUI
Professeur, EPFL / *Examinateur*
Maarten VAN STEEN
Professor, VU Amsterdam / *Examinateur*
Anne-Marie KERMARREC
Directrice de recherche, INRIA / *Directrice de thèse*
Aline ROUMY
Chargée de recherche, INRIA / *Co-directrice de thèse*

REMERCIEMENTS

Nombre de personnes m'ont accompagné durant ces trois ans qui ont mené aux travaux présentés dans ce manuscrit et j'aimerais les en remercier.

Je commencerais bien sûr par les membres de mon jury qui m'ont fait l'honneur de se pencher sur mes travaux et qui ont décidé il y a quelques jours à l'issue de ma soutenance, de me décerner le diplôme de docteur. Cet honneur est d'autant plus grand que ce sont des chercheurs et enseignants qui m'ont formé avec une grande pédagogie, inspiré tant par leur créativité que par la clarté et l'enthousiasme contagieux de leurs exposés et soutenu toutes ces années. En plus de ces qualités communes, j'ai la chance et le plaisir de pouvoir pointer du doigt pour chacun d'eux un aspect singulier de nos relations. Luc Bougé, qui a accepté de présider ce jury, m'a communiqué sa passion pour l'informatique et sa rigueur à travers ses cours d'une pédagogie rare. Je souhaite également le remercier pour son soutien constant et le temps qu'il m'a consacré. Carole Delporte-Gallet et Pablo Rodriguez, qui ont accepté de rapporter mon manuscrit, m'ont apporté des remarques et questions pertinentes sur mes travaux. D'autre part, je leur suis grandement reconnaissant pour m'avoir respectivement guidé et conseillé pour la poursuite de ma carrière et m'avoir offert l'opportunité de découvrir le passionnant domaine des réseaux et l'univers de la recherche industrielle via un stage à Telefonica. Enfin, Rachid Guerraoui et Maarten van Steen ont été pour moi des collaborateurs et encadrants précieux qui m'ont grandement inspiré. Je ne peux m'empêcher de penser que les qualités d'originalité et de rigueur que les membres du jury ont soulignées dans ma thèse sont dues à Rachid avec qui j'ai mené les travaux présentés dans ce manuscrit. Quant à Maarten van Steen, dans chacune de nos rencontres, que ce soit lors de mon semestre d'étude à l'Université libre d'Amsterdam ou lors de nos collaborations, sa vision pratique de la recherche et ses conseils ont grandement influencé le chercheur que je suis.

Ces remerciements me mènent naturellement à ma directrice de thèse, Anne-Marie Kermarrec à qui je dois, entre beaucoup d'autres choses, toutes ces rencontres. Je commencerais donc par la remercier chaleureusement de m'avoir permis de connaître ces chercheurs (et par extension je remercie Luc Bougé de me l'avoir présentée). Avoir eu Anne-Marie comme directrice a fait de cette thèse une expérience unique et je lui suis infiniment reconnaissant pour son soutien scientifique et moral. Je réalise aujourd'hui la chance qui a été la mienne de travailler à ses côtés durant ces trois années.

Enfin, je souhaite remercier Aline Roumy, qui a largement compensé la faible fréquence de nos collaborations en début de thèse (indépendante de notre volonté) par une aide précieuse en fin de thèse tant pour la soutenance que pour l'écriture du manuscrit.

Ces trois ans m'ont permis de rencontrer beaucoup de personnes que je suis heureux de connaitre aujourd'hui et qui m'ont énormément apporté. Je commencerais par Max, avec qui j'ai pris un immense plaisir à travailler et que je suis heureux de compter parmi mes amis. Les meilleurs moments

de ma thèse sont certainement ceux passés ensemble. Ionut est également un collègue et ami très cher à mes yeux. Travailler ensemble a été, est et sera un plaisir pour moi. Il serait honteux d'oublier Romain et Aude et Simon qui m'ont accompagné et soutenu avec beaucoup d'attention pendant ces trois années et bien sûr Hervé pour avoir été un mentor sans faille et un ami. Enfin, je finirais par remercier Marin, Erwan, Vincent et Nicolas pour les moments passés ensemble à l'IRISA ou ailleurs : leur présence a été très importante pour moi.

Je finirais ces remerciements par ma famille et amis de longue date : mes parents bien sûr, avec un petit plus pour mon père pour son exceptionnelle relecture, mon frère Duane et ma cousine Sarah qui ont eux aussi participé à la relecture, Ibrahim et Hassan, et bien sûr Aune, pour tout, tout simplement.

CHAPITRE 1

INTRODUCTION

Un système réparti est un ensemble d'entités capables d'effectuer des calculs et d'échanger des informations suivant un protocole dans le but d'accomplir collectivement une tâche. La conception d'un protocole pour un système donné se base sur un ensemble de règles spécifiant son comportement, le modèle. Dans le cas d'un système réparti, le modèle spécifie le comportement des entités et des liens de communication. Techniquement parlant, Internet est un réseau d'ordinateurs et, par conséquent, le modèle considéré est souvent fondé sur les caractéristiques des ordinateurs.

Les entités, tout comme les liens de communication, étant faillibles, il est important de définir également leur comportement en cas de problème. Il s'agit du modèle de *fautes*. Ce dernier joue un rôle déterminant dans l'étude du bon fonctionnement d'un système réparti. Le modèle de *pannes franches* par exemple, où une entité est supposée suivre le protocole correctement jusqu'à un instant donné où elle cesse définitivement de fonctionner, est couramment utilisé et est également fondé sur un comportement possible d'un ordinateur.

Le modèle de *fautes byzantines* traduit le cas où une entité fautive peut adopter n'importe quel comportement. En d'autres termes, aucune restriction n'est posée sur le comportement des entités fautives et ce modèle est donc, en un sens, universel. Suivant le modèle de fautes considéré, l'approche et la méthode employées pour garantir le bon fonctionnement du système varient suivant que l'on préfère empêcher les fautes ou masquer leur impact. Dans le premier cas, la cryptographie est souvent utilisée, rendant ainsi impossible certains comportements pour un certain modèle de puissance de calcul des entités. Dans le second cas, une technique classique consiste à ignorer les fautes isolées, en procédant à un vote de majorité parmi les entités pour valider chaque étape du protocole réparti.

Je considère dans ce manuscrit un nouveau modèle de fautes et propose une approche originale pour gérer ces dernières, tous deux basés sur le constat suivant : dans le cas de nombreux services répartis déployés sur Internet, des systèmes de partage de fichiers aux réseaux sociaux, l'entité est un utilisateur et non un ordinateur. Certes les interactions entre les utilisateurs se font par l'intermédiaire d'ordinateurs, mais l'utilisateur est celui qui prend les décisions. Par conséquent, le modèle de pannes franches reste pertinent mais insuffisant dans ce cas car il n'englobe pas les fautes in-

tentionnelles initiées par un utilisateur. À l'autre extrême, le modèle de fautes byzantines englobe trop largement les fautes commises par un utilisateur. Il est en effet raisonnable de penser qu'un utilisateur dévie intentionnellement d'un protocole dans un but précis et n'adopte donc pas *tous* les comportements possibles. On peut toutefois noter que le modèle de fautes byzantines peut parfois être adapté au comportement d'un utilisateur : si le programme exécutant le protocole réparti a été écrit par un utilisateur, il peut comporter des erreurs conduisant à un comportement erratique capturé par le modèle de fautes byzantines.

La première étape de ma démarche est donc de définir un modèle de *tricheur rationnel* pour des utilisateurs fautifs d'un certain service déployé sur Internet, qui se base sur ce qui est *important* pour un utilisateur. La seconde étape consiste à exploiter l'aspect rationnel des utilisateurs, représenté par une fonction d'intérêt qu'ils tentent de maximiser, pour les dissuader de mettre en œuvre certaines déviations du protocole. En d'autres termes, on fait en sorte, par le biais de sanctions, qu'une tricherie n'augmente pas la fonction d'intérêt d'un utilisateur : il reste à trouver un moyen de pression idoine.

Dans le cas où l'utilisateur participant au protocole réparti tire directement bénéfice de sa participation, la fonction d'intérêt se définit naturellement comme étant son bénéfice, et le moyen de pression consiste à exclure l'utilisateur déviant du protocole, réduisant ainsi son intérêt à zéro. Sinon, il faut utiliser un facteur extérieur sur lequel il est possible d'agir et qui intervienne dans la fonction d'intérêt de l'utilisateur. Dans les deux cas, le problème de la détection répartie des tricheries se pose.

Pendant mon doctorat, j'ai défendu l'approche dissuasive basée sur des sanctions et utilisant des informations obtenues au moyen de vérifications réparties ne faisant pas usage de méthodes cryptographiques. Ces méthodes sont illustrées dans deux contextes applicatifs, la diffusion collaborative de vidéos sur Internet et le sondage réparti dans un réseau social en ligne, correspondant respectivement aux deux cas de figure énoncés dans le paragraphe précédent.

Pour la diffusion collaborative de contenu, abordée dans le chapitre 3, un tricheur tente de diminuer sa contribution pour économiser sa bande-passante tout en assurant un visionnage de bonne qualité. L'intérêt d'un tricheur est une fonction croissante de la quantité de données reçues et une fonction décroissante de la quantité de données envoyées. La première contribution présentée dans ce manuscrit consiste à rendre robuste un protocole existant de diffusion collaborative épidémique de contenu à l'aide de méthodes dissuasives pour le modèle de tricheurs rationnels ainsi défini.

Dans le cadre des réseaux sociaux en ligne, abordés dans le chapitre 4, un facteur extérieur intervenant dans l'intérêt de tout utilisateur est sa réputation, qui peut être impactée par les informations apparaissant sur son profil. Si on prend l'exemple d'un sondage dans un réseau social, un tricheur est intéressé de connaître l'opinion des autres participants et de biaiser le résultat, mais il se soucie également de préserver sa réputation. Par conséquent, des mécanismes de détection de tricheries capables d'annoter le profil d'un utilisateur, et donc de ternir sa réputation, permettent de contrebalancer le bénéfice qu'un utilisateur retire de ses tricheries. Le prix à payer pour utiliser l'aspect personnel des réseaux sociaux est, entre autres, de fournir de fortes garanties de confidentialité. La formulation du modèle de tricheurs rationnels dans un réseau social, la formalisation du problème de calcul réparti sous ce modèle et la conception d'un protocole permettant de le résoudre constituent la seconde contribution présentée dans ce manuscrit.

Il apparaît à travers ces deux cadres d'étude que l'approche originale proposée, à savoir un modèle de tricheur rationnel exploité par des méthodes de vérifications réparties conduisant à des sanctions, et défendue dans ce manuscrit permet de concevoir des *systèmes pratiques* résistants aux tricheurs

mais également d'envisager l'*étude théorique* des systèmes répartis sous un angle inédit et rendu possible par l'émergence des réseaux sociaux en ligne.

Contributions annexes

Au cours de mon doctorat, j'ai également eu l'occasion de participer à des projets de recherche qui ne s'inscrivent pas dans l'optique des méthodes dissuasives pour lutter contre les utilisateurs malhonnêtes dans les systèmes répartis présentées dans ce manuscrit, et ne sont par conséquent pas détaillés dans le manuscrit. Ces travaux annexes concernent également les systèmes répartis à grande échelle et sont décrits brièvement dans cette section.

Déploiement spontané de réseaux d'actionneurs mobiles sans fil, avec Éric Fleury (ENS Lyon)

L'objectif de ce projet est d'étudier la relation entre le routage et le déploiement spontané dans un réseau d'actionneurs sans fil dont la mobilité est contrôlée. On considère le problème d'action à la demande où des entités, déployées dans une région d'intérêt, doivent effectuer des actions à des points précis de la région. Les ordres émanent des entités qui peuvent alors (1) transmettre l'ordre à une autre entité, (2) se déplacer ou (3) exécuter l'action elles-mêmes, si c'est à leur portée. Dans ce contexte applicatif, on présente Grasp : un protocole de routage décentralisé qui combine les options (1), (2) et (3) de sorte qu'une des entités du réseau finisse par exécuter l'action demandée. Sous certaines conditions sur le nombre d'entités (par rapport à la taille de la zone d'intérêt) et sur l'origine des ordres, on montre qu'un ensemble d'entités exécutant Grasp se déploie spontanément dans la zone d'intérêt de telle sorte qu'un ordre émanant de n'importe laquelle d'entre elles peut être exécuté sans qu'aucune entité n'ait à se déplacer, assurant ainsi une couverture connectée de la zone. Grasp apparaît dans les actes de la conférence DCOSS 2009 (6th IEEE International Conference on Distributed Computing in Sensor Systems) [8].

Vidéo à la demande compatible avec l'approche donnant-donnant, avec Vivek Rai (VU Amsterdam) et Maarten van Steen (VU Amsterdam)

L'objectif de ce projet est de concevoir un système pair-à-pair collaboratif de vidéo à la demande compatible avec le mécanisme incitatif, dit donnant-donnant, utilisé par BitTorrent. Dans un système traditionnel de vidéo à la demande, les blocs composant la vidéo sont téléchargés dans l'ordre, ce qui pose la question suivante : comment deux utilisateurs à des positions différentes dans la vidéo peuvent tous les deux avoir un bloc qui intéresse l'autre ? Le protocole proposé résout ce problème en combinant et en exploitant au maximum des techniques de téléchargement anticipé de blocs (qui ne sont pas utiles immédiatement) et de regroupement des blocs consécutifs en segments (à l'intérieur desquels la contrainte de séquentialité du téléchargement est relaxée), à l'aide d'une structure souple maintenue simplement par commérage. Ce travail apparaît dans les actes de la conférence NOSSDAV 2010 (20th International Workshop on Network and Operating Systems Support for Digital Audio and Video) [9].

Codes réseau LT, avec Mary-Luc Champel (Technicolor R&D) et Nicolas Le Scouarnec (Technicolor R&D)

L'objectif de ce projet est de concevoir des codes réseau de faible complexité pour la diffusion collaborative de contenu. Le codage réseau est un paradigme de communication consistant à combiner les données reçues avant de les transmettre permettant ainsi d'atteindre le flot maximal d'un graphe. Sous sa forme actuelle, c'est-à-dire le codage réseau linéaire aléatoire (RLNC pour *Random Linear Network Coding*), le codage réseau donne des résultats optimaux en terme de rapidité de diffusion au prix d'une forte complexité de décodage de l'ordre de $\mathcal{O}(k^2)$ opérations élémentaires par bit de donnée, où k est le nombre de blocs du contenu. Les codes réseau proposés, à savoir LTNC (pour *Luby Transform Network Codes*), sont basés sur la transformation de Luby (LT) qui permet de construire des codes sous-optimaux pouvant être décodés en $\mathcal{O}(k \log k)$ opérations élémentaires par bit de donnée par *propagation de croyance* (pour *belief propagation*). Le cœur du codeur réseau LTNC est composé d'un ensemble d'heuristiques permettant de combiner des blocs encodés en préservant la structure spécifiée par LT. LTNC a été présenté sous forme d'un poster à la conférence CoNEXT 2009 (5th International Conference on emerging Networking EXperiments and Technologies) [1] et il apparaît dans les actes de la conférence ICDCS 2010 (30th International Conference on Distributed Computing Systems) [2] où il a obtenu le *prix du meilleur article*.

Publications

[1] Mary-Luc CHAMPEL, Kévin HUGUENIN, Anne-Marie KERMARREC et Nicolas LE SCOUARNEC : LT Network Codes. *In CoNEXT'09: Proceedings of the 5th ACM International Conference on emerging Networking EXperiments and Technologies (Student Workshop)*, 2009. Extended abstract.

[2] Mary-Luc CHAMPEL, Kévin HUGUENIN, Anne-Marie KERMARREC et Nicolas LE SCOUARNEC : LT Network Codes. *In ICDCS'10: Proceedings of the 30th IEEE International Conference on Distributed Computing Systems*, 2010. Best paper award.

[3] Andrei GIURGIU, Rachid GUERRAOUI, Kévin HUGUENIN et Anne-Marie KERMARREC : Computing in Social Networks. *In SSS'10: Proceedings of the 12th International Symposium on Stabilization, Safety and Security of Distributed Systems*, 2010.

[4] Rachid GUERRAOUI, Kévin HUGUENIN, Anne-Marie KERMARREC, Maxime MONOD et Swagatika PRUSTY : LiFTinG: Lightweight Freerider-Tracking Protocol in Gossip. *In MIDDLEWARE'10: Proceedings of the 11th ACM/IFIP/USENIX International Middleware Conference*, 2010.

[5] Rachid GUERRAOUI, Kévin HUGUENIN, Anne-Marie KERMARREC et Maxime MONOD : Decentralized Polling with Respectable Participants. *In OPODIS'09: Proceedings of the 12th International Conference on Principles of Distributed Systems*, 2009. Best student paper award.

[6] Rachid GUERRAOUI, Kévin HUGUENIN, Anne-Marie KERMARREC et Maxime MONOD : On Tracking Freeriders in Gossip Protocols (Short paper). *In P2P'09: Proceedings of the 9th IEEE International Conference on Peer to Peer Computing*, 2009. Short paper.

[7] Rachid GUERRAOUI, Kévin HUGUENIN, Anne-Marie KERMARREC et Maxime MONOD : Towards Secured Distributed Polling in Social Networks. *In DISC'09: Proceedings of the 23rd International Symposium on Distributed Computing*, 2009. Brief announcement.

[8] Kévin HUGUENIN, Anne-Marie KERMARREC et Eric FLEURY : Route in Mobile WSN and Get Self-Deployment for Free. *In DCOSS'09: Proceedings of the 5th International Conference on Distributed Computing in Sensor Systems*, 2009.

[9] Kévin HUGUENIN, Anne-Marie KERMARREC, Vivek RAI et Maarten van STEEN : Designing a Tit-for-Tat Based Peer-to-Peer Video-on-Demand System. *In NOSSDAV'10: Proceedings of the 20th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2010.

CHAPITRE 2

TRAVAUX CONNEXES

Ce chapitre décrit les travaux fondateurs liés au sujet de recherche abordé dans ce manuscrit, à savoir la modélisation d'utilisateurs rationnels dans le cadre des systèmes collaboratifs et du calcul dans les réseaux sociaux. En premier lieu, on détaille les modèles traditionnels de fautes utilisés pour les systèmes répartis ainsi que les approches classiques pour fournir des garanties dans ces modèles. Puis on décrit la théorie des jeux qui formalise la notion d'utilisateur rationnel, ainsi que les travaux notables qui s'en sont inspirés pour des problématiques concernant les systèmes collaboratifs. On présente ensuite les modèles de calcul utilisés pour les systèmes répartis ainsi que les caractérisations de la confidentialité, aspect incontournable des réseaux sociaux en ligne.

2.1 Modèles de fautes

Dans [AL86], Avižienis et Laprie passent en revue les concepts et la terminologie du calcul fiable. En particulier ils présentent une taxinomie des modèles de fautes et des méthodes pour gérer les fautes en général. On recense trois modèles de fautes liés aux travaux présentés dans ce manuscrit : le modèle de *pannes franches*, où une entité fautive cesse *définitivement* de fonctionner, le modèle de *pannes transitoires*, où une entité fautive cesse *temporairement* de fonctionner et les *pannes par omission*, où des messages sont perdus. Comme noté dans l'introduction, ces modèles de fautes sont adaptés aux entités (à savoir les ordinateurs pour les deux premiers) et aux liens de communication (à savoir des canaux de communication bruités pour le dernier) d'un réseau informatique. À ces modèles, s'ajoute celui des *fautes byzantines*, introduit par Lamport *et al.* [LSP82], dans le problème de l'attaque coordonnée : N commandements de l'armée byzantine, chacun dirigé par un général, campent autour d'une cité ennemie et doivent, pour remporter la victoire, tous attaquer en même temps, en sachant que certains généraux sont des traîtres et que les messagers peuvent être interceptés et le contenu des messages qu'ils transportent modifié. De manière plus générale, le modèle de faute byzantine spécifie que les entités fautives peuvent adopter un comportement *arbitraire*. Avižienis et Laprie recensent quatre méthodes pour gérer les fautes et ainsi concevoir un système fiable : supprimer les fautes, prévoir les fautes, empêcher les fautes et tolérer les fautes. La première

approche consiste à s'assurer par exemple que le matériel utilisé est fiable et que les programmes mettant en œuvre le protocole sont justes. La seconde consiste à estimer l'impact des fautes et à le corriger. On décrit plus en détail les deux dernières approches dans les paragraphes suivants.

Un exemple typique de technique d'empêchement de fautes est la *cryptographie*. On décrit rapidement deux types de méthodes cryptographiques basées sur des *clés*, c'est-à-dire une suite de bits prise en paramètre par les primitives cryptographiques. La cryptographie symétrique (AES et DES par exemple) consiste à chiffrer, à l'aide d'une clé, les données échangées de telle sorte que seules les entités possédant la clé sont capables de les déchiffrer, assurant ainsi la confidentialité. Combinée avec une méthode de *code d'authentification de message* (MAC, pour *Message Authentication Code*), cette méthode de chiffrement peut garantir l'intégrité des données en permettant aux destinataires de vérifier que les données reçues sont bien celles envoyées. La cryptographie asymétrique (RSA par exemple) suppose que chaque entité possède une clé privée connue par elle seule et une clé publique connue par tout le monde et fournit deux primitives : le chiffrement et la signature. Un message destiné à une entité est chiffré avec sa clé publique et ne peut être déchiffré qu'avec sa clé privée. Un message émis par une entité est signé avec sa clé privée et on peut vérifier la validité de la signature avec sa clé publique. La cryptographie asymétrique nécessite une infrastructure annexe pour diffuser les clés publiques des entités (PKI pour *Public Key Infrastructure*). Le fait que certaines opérations ne puissent être effectuées qu'avec la connaissance d'une clé constitue effectivement un empêchement de fautes. Cependant, cette propriété repose sur l'hypothèse qu'il n'existe pas de manière *efficace* de résoudre certains problèmes mathématiques (la factorisation de grands entiers pour l'algorithme RSA par exemple).

La tolérance aux fautes, potentiellement byzantines, est une approche consistant à *masquer* l'impact de celles-ci. Lorsqu'une tâche est assurée par une unique entité, le résultat est compromis dès lors que celle-ci est fautive. L'approche traditionnelle est de faire exécuter la tâche simultanément à plusieurs entités et d'effectuer un vote sur le résultat. Chaque entité propose son résultat et le résultat le plus représenté est choisi. Dans le modèle de pannes, un système réparti peut tolérer jusqu'à $N - 1$ entités fautives simultanément, car l'entité correcte étant la seule restante, son résultat, qui est correct, est élu. Dans le modèle de fautes byzantines, les entités fautives peuvent former une coalition et décider de donner le même résultat erroné. Par conséquent, pour tolérer un tel comportement, les entités correctes doivent former une majorité du système, soit un total d'au moins $2f + 1$ entités où f est le nombre maximum d'entités simultanément fautives.

PeerReview [HKD07] est un des systèmes les plus aboutis permettant de détecter de manière efficace et sûre les fautes byzantines des entités dans un protocole déterministe quelconque. Il ne constitue pas en lui-même un mécanisme d'empêchement de fautes byzantines mais il empêche les entités de dissimuler leur fautes. PeerReview utilise des méthodes cryptographiques et se base sur le principe de *responsabilité* (*accountability* en anglais) : chaque entité conserve un historique de ses actions et interactions passées. Lors d'une vérification, les fautes sont détectées par une simulation utilisant une implémentation de référence du protocole. De cette manière PeerReview garantit que les fautes byzantines ayant un impact sur les entités correctes sont détectées à coup sûr et qu'une entité correcte peut prouver qu'elle n'est pas fautive, empêchant ainsi les accusations à tort. La correction de PeerReview repose sur l'hypothèse que les messages ne sont pas perdus. La complexité en message est en $\mathcal{O}(N^2)$ mais peut être réduite à $\mathcal{O}(\log N)$ si on accepte que les garanties soient assurées de manière probabiliste. Le système CSAR [BDHU09] permet d'étendre l'utilisation de PeerReview aux protocoles non-déterministes à l'aide de générateurs cryptographiques de nombres aléatoires tout en garantissant que le comportement des entités reste imprévisible aux

yeux des autres, ce qui assure la confidentialité de leurs actions futures. PeerReview est le système pratique utilisant la cryptographie pour gérer les comportement byzantins le plus proche de l'approche proposée dans ce manuscrit. On notera cependant que bon nombre d'autres systèmes pratiques ont été proposés en commençant par [CL02] et jusqu'aux récents Zyzzyva [KAD+09], Aardvark [CWA+09] et Aliph [GKQV10].

2.2 Théorie des jeux

La théorie des jeux, introduite par van Neumann et Morgenstern [NM44] pour l'économie, spécifie le comportement d'individus rationnels (supposé modéliser le comportement humain) et constitue donc une source d'inspiration pour le travail présenté dans ce manuscrit. De manière générale, le contexte applicatif de la théorie des jeux est un ensemble d'individus qui interagissent. Les actions qu'ils peuvent exécuter et l'effet de celles-ci sur les individus du système sont spécifiés par un ensemble de règles. L'objectif d'un individu consiste à maximiser une certaine fonction d'intérêt, numérique ou pas, et il dispose pour atteindre cet objectif d'un ensemble de stratégies. La complexité du problème vient du fait que les intérêts des individus sont liés et une action de l'un d'eux peut affecter les autres. Par conséquent la stratégie d'un individu dépend de celles des autres et peut donc changer au cours du temps pour s'y adapter. La notion d'équilibre, c'est-à-dire une situation où tous les individus n'ont pas intérêt à changer leur stratégie, est un concept clé de la théorie des jeux et a été largement étudiée. On peut par exemple citer l'équilibre de Nash pour le cas d'individus ne formant pas de coalition [Nas50, Nas51]. Il est souvent supposé que si un tel équilibre existe les individus tendent à appliquer la stratégie correspondante [Mai98].

Le mécanisme incitatif dit du *donnant-donnant* (*Tit-for-tat* en anglais) utilisé dans BitTorrent est une application directe de la théorie des jeux aux systèmes répartis de partage collaboratif de fichiers. Dans BitTorrent, le fichier partagé est découpé en blocs qui fixent la granularité des échanges entre les individus. L'intérêt d'un individu y est défini comme une fonction du nombre de blocs qu'il reçoit et du nombre qu'il envoie. Le premier quantifie directement le bénéfice de l'individu et le second le coût. Le mécanisme du donnant-donnant consiste à forcer un individu à envoyer un bloc en échange de chaque bloc qu'il reçoit. De cette manière, le protocole de partage de fichiers devient optimal au sens de Pareto, c'est-à-dire que chaque échange de blocs augmente l'intérêt des deux protagonistes, car le coût d'un envoi est supposé inférieur au bénéfice engendré par la réception d'un bloc. Ce mécanisme est dit incitatif. Par conséquent, le système converge vers une situation où les individus ne peuvent plus augmenter leur bénéfice, c'est-à-dire où ils ont tous reçu le fichier intégralement. On peut observer plusieurs points : un individu n'ayant aucun bloc ne peut pas participer et les individus ayant reçu le fichier intégralement, au même titre que l'individu qui introduit le fichier dans le système (appelé source), n'ont en fait aucun intérêt à envoyer des blocs aux autres individus du système. Dans BitTorrent ce problème est résolu par l'attitude altruiste des individus ayant reçu l'intégralité du fichier, c'est-à-dire qu'ils envoient des blocs sans rien attendre en retour. Ce comportement altruiste est capturé par le modèle BAR [AAC+05a] que l'on présente dans la suite. On note qu'une stratégie rationnelle consiste à exploiter les individus altruistes [PIA+07, SPCY07].

Bon nombre de travaux utilisent de manière informelle les résultats de la théorie des jeux [CN03, SPYJ07]. On peut citer à titre d'exemple le protocole de sauvegarde collaborative de contenu Samsara [CN03] qui généralise le concept du donnant-donnant en utilisant des reconnaissances de dettes. Lorsqu'un individu héberge du contenu pour un autre et que ce dernier n'est pas capable d'en héberger, il donne en échange une reconnaissance de dette signée au moyen de primitives cryptogra-

phiques qui peuvent alors être échangées formant ainsi des chaînes de dettes. Le système proposé met en œuvre efficacement ce mécanisme de crédit en annulant les cycles dans les chaînes. Les individus qui ne respectent pas leurs dettes sont sanctionnés en supprimant une partie de leurs données des endroits où elles sont sauvegardées pour annuler leurs dettes. On note que si l'on fait abstraction de la dimension philosophique du problème, un mécanisme de sanctions est similaire à un mécanisme incitatif : d'un point de vue technique, sanctionner un mauvais comportement en diminuant l'intérêt de l'individu fautif revient à récompenser un comportement augmentant l'intérêt des autres individus. La menace de supprimer le contenu qu'un individu a sauvegardé constitue un exemple de moyen de pression direct permettant de diminuer l'intérêt d'un individu, l'intérêt étant ici une fonction croissante du nombre de blocs qu'un individu sauvegarde chez les autres. De même que pour BitTorrent, la capacité de stockage prêtée par un individu au système représente un coût et impacte donc négativement son intérêt. Dans le même esprit, le protocole Dandelion [SPYJ07] utilise, dans le cadre de diffusion de vidéos à la demande, un système de crédits gérés par une entité centrale et pouvant être convertis sous forme monétaire ou pour accroître l'intérêt d'un individu, en augmentant par exemple le nombre de vidéos pouvant être visionnées et la rapidité du téléchargement.

Le modèle BAR (pour byzantin, altruiste ou rationnel) [AAC+05a] propose une modélisation plus complète des utilisateurs d'Internet en combinant à la fois l'aspect rationnel de ces derniers, en se basant sur la théorie des jeux, et les comportements erratiques des ordinateurs. Le modèle altruiste permet de capturer le comportement d'un individu tel que la source évoquée dans le cas de BitTorrent. Plus précisément les trois modèles d'individus sont définis de la manière suivante. Les individus altruistes suivent le protocole sans se soucier de leur intérêt, ce qui correspond au modèle de nœud honnête dans le domaine de la tolérance aux fautes. Les individus rationnels cherchent à maximiser leur fonction d'intérêt et ne forment pas de coalition. Si un équilibre de Nash existe pour la fonction d'intérêt considérée, ils sont supposés adopter la stratégie correspondante. Enfin, les individus byzantins dévient du protocole de manière arbitraire. Dans ce modèle, deux classes de protocoles sont définies : celle des systèmes dans lesquels il est de l'intérêt des individus rationnels de suivre le protocole (appelés systèmes incitatifs) et les systèmes qui fonctionnent correctement en présence de toute déviation rationnelle. Le modèle BAR suggère également d'identifier les utilisateurs de manière à pouvoir appliquer des sanctions indépendantes du protocole. C'est précisément cette démarche qu'adopte le travail présenté dans le chapitre 4 de ce manuscrit en exploitant, par le biais de la réputation, l'appartenance des entités du système à un réseau social.

Les protocoles BAR Gossip [LCW+06] et FlightPath [LCM+08] garantissent la diffusion efficace de contenu dans le modèle BAR présenté dans le paragraphe précédent. BAR Gossip, comme son nom l'indique, repose sur un algorithme de diffusion, dit épidémique, où un individu choisit périodiquement, uniformément au hasard, d'autres individus du système, appelés partenaires, avec qui échanger des blocs. Les échanges de blocs s'effectuent en suivant le principe du donnant-donnant de BitTorrent. Tout comme PeerReview, BAR Gossip enregistre l'historique des actions des individus dont l'authenticité est certifiée à l'aide de primitives cryptographiques. Il est alors possible de vérifier lors de l'audit d'un individu qu'il a correctement suivi le protocole. BAR Gossip résout le problème des individus envoyant des blocs invalides, générés aléatoirement par exemple, pour pouvoir recevoir des blocs d'un individu à qui il ne peuvent fournir aucun bloc de la manière suivante : les blocs envoyés sont chiffrés par l'expéditeur à l'aide d'une primitive cryptographique utilisant une clé, qui n'est transmise que lorsque l'intégrité des blocs reçus en échange a été vérifiée. Ainsi, dans un échange, le bénéfice d'un individu est conditionné par le bénéfice de l'autre. Les protocoles épidémiques étant aléatoires ils sont a priori imprévisibles ce qui rend la vérification de la

sélection des partenaires difficile. BAR Gossip, tout comme CSAR, résout ce problème à l'aide de générateurs de nombres pseudo-aléatoires. La vérification nécessite donc uniquement la connaissance de la graine du générateur dont l'intégrité est assurée à l'aide de primitives cryptographiques. Les vérifications mises en œuvre dans BAR Gossip conduisent à l'émission de *preuves de tricherie* (POM pour *proof of misbehavior* en anglais) utilisées pour décider de l'exclusion d'un individu. Le protocole FlightPath s'inscrit dans la même optique que BAR Gossip mais il repose sur la notion d'équilibre approximé, ou ε-équilibre, qui incorpore le coût d'une déviation. Concrètement, mettre en œuvre une déviation au protocole consiste à installer une version alternative du programme implémentant le protocole. Une déviation est alors considérée rentable, et donc mise en œuvre par un individu rationnel, si l'intérêt est au moins multiplié par $(1 + \varepsilon)$.

2.3 Modèles de calculs

Le *modèle de calcul* utilisé pour les entités est également crucial dans la conception et l'analyse d'un système réparti. Il définit d'une part la puissance des entités fautives et donc de leur fautes mais également les moyens dont disposent les entités non-fautives pour s'en protéger. Plus précisément, le modèle de calcul spécifie les opérations que peuvent effectuer les entités, la mémoire à leur disposition, le nombre de messages qu'elles peuvent envoyer, *etc.*

Le modèle de *protocole de population* [AAD$^+$06], proposé par Angluin *et al.*, fournit un cadre théorique pour l'étude des systèmes composés d'entités mobiles avec des capacités limitées. Dans ce modèle, chaque entité dispose d'une mémoire en $\mathcal{O}(1)$. Le calcul est réalisé par une succession d'interactions deux-à-deux entre les entités se produisant lorsque deux entités sont suffisamment proches. Les rencontres entre les entités sont contrôlées par un ordonnanceur. Les résultats et les performances des protocoles de population dépendent des propriétés de l'ordonnanceur, par exemple le fait que deux entités quelconques puissent se rencontrer (graphe d'interaction complet) et se rencontrent infiniment souvent (équité). Un protocole de population fait intervenir un alphabet d'entrée X, un ensemble d'états Q et une fonction de transition $\delta : Q \times Q \to Q \times Q$. À un instant donné, une entité se trouve dans un état de Q. L'état initial d'une entité est déterminé par sa valeur d'entrée. Lors d'une interaction, les deux protagonistes changent d'état en fonction de leur état actuel, comme spécifié par la fonction de transition δ. On dit qu'un protocole de population calcule une certaine fonction (dont la valeur d'entrée est codée par les entrées des entités) s'il converge vers une configuration où les états des entités codent le résultat de la fonction. Il a été montré que les protocoles de population peuvent calculer les prédicats définis dans l'arithmétique de Presburger du premier ordre, c'est-à-dire les fonctions qui s'écrivent comme une combinaison de formules booléennes par les opérateurs logiques $\wedge$, $\vee$ et $\neg$ et les quantificateurs $\exists$, $\forall$. Les formules booléennes sont elles-mêmes construites à l'aide des opérateurs $=$, $\leq$ et $\geq$ sur des additions d'entiers. Par exemple, la fonction $(x \leq y + 2) \vee (w \geq y - 2)$ fait partie des fonctions calculées.

Les résultats présentés dans cet article ont été étendus au niveau de la puissance de calcul en relaxant l'hypothèse d'interactions deux-à-deux [AAER07] et l'hypothèse de graphe d'interaction complet [AAC$^+$05b]. Les aspects de tolérance aux pannes sont abordés dans [DGFGR06] en simulant plusieurs exécutions simultanées puis en effectuant un vote sur les résultats de ces dernières. La méthode proposée est générique et consiste à transformer n'importe quel protocole de population pour le rendre résistant aux pannes transitoires ou définitives. Les différentes simulations sont construites de telle sorte qu'une faute a un impact sur au plus deux d'entre elles. La problématique de la confidentialité des entrées des entités est abordée dans [DGFGR07] en proposant une méthode

de partage de secret. Ce travail considère le cas d'un attaquant passif qui essaie de découvrir la valeur initiale des autres entités sans dévier du protocole. La méthode proposée consiste à partager l'entrée d'une entité et à la diffuser en plusieurs fois. Dans le cas où les valeurs d'entrée des entités sont des entiers et que la fonction calculée est la somme modulo m, pour un m donné, une entité ayant une entrée i peut agir comme si son entrée valait 1 pour les i prochaines rencontre, évitant ainsi de communiquer la valeur de son entrée à la première entité qu'elle rencontre.

Le modèle de *protocoles de communauté* [GR09], proposé par Guerraoui et Ruppert, dote chaque entité d'un identifiant unique. Ce modèle étend également la mémoire des entités de sorte qu'elles puissent stocker $\mathcal{O}(1)$ identifiants. Au prix de cette hypothèse supplémentaire, les protocoles de communauté peuvent simuler l'exécution d'une machine de Turing et par conséquent ils peuvent résoudre les problèmes de décision dans $\mathrm{NSPACE}(N \cdot \log N)$. De plus ils peuvent tolérer un nombre constant $f = \mathcal{O}(1)$ de fautes byzantines dès lors que le résultat de la fonction calculée reste inchangé si moins de f caractères d'entrée sont corrompus et moins de $3f + 1$ sont supprimés. La complexité en temps et en message des protocoles de communauté, comme pour les protocoles de population, est généralement polynomiale en N où N désigne le nombre d'entités.

2.4 Confidentialité

L'émergence des réseaux sociaux en ligne, du vote électronique et des bases de données contenant des informations personnelles a motivé un grand nombre de recherches. Pour pouvoir concevoir des mécanismes de protection de la vie privée des utilisateurs il faut définir en amont ce qu'est la confidentialité. On présente dans cette section des caractérisations de la confidentialité ainsi que des travaux de recherche pour la protéger.

La première approche pour définir la confidentialité dans le cadre du calcul consiste à dire qu'un protocole, réparti ou pas, est confidentiel s'il ne révèle pas plus d'information sur les entrées que le résultat du calcul lui-même. Dans le cas d'un vote électronique dont le résultat serait 60% pour le premier candidat contre 40% pour le second, on doit garantir que la seule information sur les votes connue par le système de comptage est qu'il y a 60% de chance qu'un utilisateur donné ait voté pour le premier candidat (si on fait abstraction des informations obtenues par d'autres moyens). Cette caractérisation est une application directe d'un principe appliqué aux primitives cryptographiques : la version chiffrée d'un message ne doit pas apporter plus d'information que ce qui est connu *a priori* par l'attaquant. Il existe de nombreux algorithmes de vote électronique garantissant la confidentialité ainsi définie à l'aide de primitives cryptographiques.

Les protocoles de vote sont des cas particuliers des protocoles de *calcul multi-utilisateur* (MPC, pour *multi-party computation*) cryptographique sécurisé, initialement proposés par Yao [Yao82] pour résoudre le problème suivant : deux millionnaires veulent savoir qui est le plus riche des deux sans révéler la fortune de chacun. Ces protocoles s'appuient sur des *méthodes cryptographiques par homomorphisme* [RAD78] (par exemple les systèmes de chiffrement RSA, de Benaloh [Ben94], de Palier [Pai99], *etc.*) dont le principe est le suivant. Les entrées des entités sont chiffrées de telle sorte qu'elles ne puissent être décodées que conjointement par l'ensemble des entités. D'autre part, et c'est là qu'apparaît la propriété d'homomorphisme, l'opération à effectuer sur les entrées peut être directement appliquée sur les versions chiffrées des entrées donnant un message qui, une fois déchiffré, est égal à la combinaison des entrées. Cela signifie que $\mathcal{E}(e_1) \otimes \mathcal{E}(e_2) = \mathcal{E}(e_1 \oplus e_2)$. Ces mécanismes, par ailleurs, permettent de garantir l'intégrité des données. Certains protocoles de vote électronique, tel que [MMP02] proposé par Malkhi qui utilise les vecteurs de vérifications améliorés

(*enhanced check vectors*), reposent sur des techniques algébriques mais ne relèvent pas réellement de la cryptographie.

Les protocoles de calcul multi-utilisateur sécurisé existent également sous une forme n'utilisant pas la cryptographie. Cette méthode, appelée partage de secret par homomorphisme dérivant de la théorie de l'information, a été introduite par Shamir [Sha79], puis étoffée par Benaloh [Ben86]. Le principe consiste à partager son entrée et distribuer les « morceaux » aux autres participants du calcul de telle sorte qu'une fois tous les morceaux de toutes les entités combinés, on peut retrouver le résultat du calcul sans dévoiler les entrées des entités. Dans la proposition initiale de Shamir, illustrée dans cas du calcul de l'addition d'entiers, l'entrée e_i de l'entité i est d'abord transformée en un polynôme dont tous les coefficients sont choisis au hasard sauf le terme constant qui est égal à son entrée : $P_i[X] = a_{i,k-1}X^{k-1} + \cdots + a_{i,1}X + e_i$, avec $k \leq N$. On a donc $P_i(0) = e_i$. Pour calculer la somme des entrées des entités, chacune d'entre elles envoie la valeur $P_i(j)$ à l'entité j. Chaque entité i somme les valeurs qu'elle reçoit, obtenant ainsi la valeur v_i du polynôme somme en i, soit $v_i = \sum_j P_j(i) = (\sum_j P_j)(i)$, qu'elle transmet ensuite à toutes les autres entités. Chaque entité connaît donc la valeur du polynôme somme en $N > k-1$ points et peut donc le reconstruire à l'aide de l'interpolation de Lagrange, par exemple. En évaluant le polynôme somme en 0, chaque entité calcule alors la somme des entrées des entités. Pour $k < N$, la propriété de confidentialité est plus faible, car k entités malhonnêtes formant une coalition peuvent découvrir les entrées des entités mais, en contrepartie, elle permet d'effectuer des vérifications sur l'intégrité des entrées agrégées. Ce concept, appelé partage de secret *vérifiable* (VSS pour *Verifiable Secret Sharing*) [CGMA85], est largement utilisé pour se prémunir contre les attaquants actifs [RBO89, Sch99] qui s'autorisent à dévier du protocole pour arriver à leurs fins. Les protocoles de MPC, cryptographiques ou pas, ont une complexité, en temps et en message, au mieux linéaire en le nombre de participants. On note que des travaux récents proposent d'utiliser les relations de confiance dans les réseaux sociaux en ligne pour optimiser l'utilisation du partage de secret [VABD09].

Les propriétés de k-anonymat (k-*anonimity* en anglais) [Sam01, Swe02] et de confidentialité différentielle (*differential privacy* en anglais) [Dwo06] caractérisent la confidentialité des résultats de requêtes sur des bases de données contenant des informations personnelles ou sensibles. Dans ce contexte, l'attaquant est l'utilisateur effectuant la requête et non les entités y répondant. Les travaux qui ont conduit à la définition du k-anonymat ont été motivés par une étude montrant qu'en recoupant les informations médicales, prétendument anonymisées et vendues par une compagnie d'assurances, et le registre des votants de Cambridge (Massachusetts, États-Unis), il était possible de connaître les pathologies de certains citoyens. La propriété des bases de données (et des attributs qu'elles contiennent) permettant d'effectuer l'association entre deux entrées est qu'il existe un unique individu ayant une certaine combinaison de valeurs pour un ensemble d'attributs présents dans les deux bases. Par exemple, les données médicales ne contiennent pas les noms des patients mais contiennent la date et l'année de naissance, le code postal et le sexe, qui apparaissent également dans le registre des votants, avec le nom cette fois. Partant de cette observation, le k-anonymat spécifie naturellement que toute combinaison de valeurs pour un ensemble d'attributs, correspond à au moins k entrées dans la base de données. La technique traditionnelle pour garantir cette propriété consiste à supprimer des attributs, ou une partie d'un attribut, dans la réponse à la requête, comme l'illustre l'exemple du tableau 2.1. On note que cette propriété révèle quand même des informations sur les individus et ne tient pas compte de celles obtenues par d'autres moyens. Par exemple, dans le cas de la base de données du tableau 2.1, en apprenant par le registre de vote qu'un certain homme (un voisin par exemple) de type asiatique est né en 1984 on peut en conclure qu'il

est soit obèse soit alcoolique, ce qui est en soit une atteinte à la vie privée. De plus, si on sait que cet individu n'est pas obèse on peut en conclure qu'il est alcoolique. On peut citer à titre d'anecdote l'annulation du concours de recommandation de contenus, initié par la société de diffusion de vidéos Netflix, car la base de données anonymisée des visionnages des utilisateurs,à laquelle les concurrents avaient accès, révélaient des informations personnelles une fois recoupée avec IMDb (*internet movie database*). La notion de confidentialité différentielle, introduite par Dwork [Dwo06],

Type	**Année**	**Sexe**	**CP**	**Pathologie**
Caucasien	1985	h	3515*	épilepsie
Caucasien	1985	h	3516*	alcoolisme
Caucasien	1985	f	3523*	hypertension
Caucasien	1985	f	3524*	hypertension
Caucasien	1984	f	3526*	obésité
Caucasien	1984	f	3527*	alcoolisme
Asiatique	1984	h	3525*	alcoolisme
Asiatique	1984	h	3521*	obésité
Asiatique	1984	h	3528*	épilepsie
Asiatique	1987	h	3529*	alcoolisme
Asiatique	1987	h	3529*	alcoolisme

(a) Données non-anonymisées

Type	**Année**	**Sexe**	**CP**	**Pathologie**
Caucasien	1985	h	351**	épilepsie
Caucasien	1985	h	351**	alcoolisme
Caucasien	1985	f	352**	hypertension
Caucasien	1985	f	352**	hypertension
Caucasien	1984	f	352**	obésité
Caucasien	1984	f	352**	alcoolisme
Asiatique	1984	h	352**	alcoolisme
Asiatique	1984	h	352**	obésité
Asiatique	1984	h	352**	épilepsie
Asiatique	1987	h	352**	alcoolisme
Asiatique	1987	h	352**	alcoolisme

(b) Données anonymisées

TABLEAU 2.1 : Anonymisation d'une base de données médicale : en supprimant les deux derniers chiffres du code postal on garantit le 2-anonymat.

caractérise la confidentialité pour une fonction quelconque, potentiellement aléatoire, appliquée à un ensemble d'entrées. Le principe différentiel de cette propriété consiste à garantir qu'une petite variation de l'ensemble des données ne change que très peu le résultat. Plus formellement, un protocole aléatoire de calcul $\mathcal{K}$ garantit l'ε-confidentialité différentielle si pour tous multi-ensembles de données E_1 et E_2 ne différant que d'un élément et pour tout sous-ensemble S de l'ensemble d'arrivée de $\mathcal{K}$, on a :

$$\mathbb{P}\left[\mathcal{K}(E_1) \in S\right] \leq \exp\left(\varepsilon\right) \times \mathbb{P}\left[\mathcal{K}(E_2) \in S\right]$$

On note que cette définition ne capture pas le cas où enlever un élément à l'ensemble d'entrée réduit à 0 la probabilité d'obtenir un résultat donné. Par exemple, si $\mathcal{K}$ calcule l'élément ayant la plus petite multiplicité dans un multi-ensemble et qu'un élément de multiplicité 1 est supprimé de E_1 alors la probabilité que le calcul renvoie cet élément devient nulle. Pour cette raison, on ajoute souvent un paramètre additif δ dans la définition de la confidentialité différentielle. La méthode traditionnelle pour obtenir l'ε-confidentialité différentielle consiste à ajouter un bruit exponentiel au résultat du calcul. La variance σ^2 du bruit à ajouter est une fonction de ε et de la *sensibilité* $\Delta\mathcal{K}$ de la fonction calculée qui quantifie l'impact d'une modification d'une entrée sur le résultat : $\sigma = \Delta\mathcal{K}/\varepsilon$ où $\Delta\mathcal{K} = \max_{E_1,E_2} \|\mathcal{K}(E_1 - \mathcal{K}(E_2)\|_1$ pour E_1 et E_2 différant d'au plus un élément.

Le protocole Airavat [RSK+10], proposé par Roy *et al.*, est un système pratique d'association/réduction (*MapReduce* en anglais) permettant d'effectuer des calculs de manière sécurisée et privée, au sens de la confidentialité différentielle. Dans Airavat, la confidentialité différentielle est garantie en ajoutant du bruit aux réponses fournies par le système. La méthode de bruitage du résultat est associée à des mécanismes de sécurité complémentaires tel que la limitation du nombre de requêtes par utilisateur (la moyenne du bruit ajouté étant nulle, effectuer un grand nombre de

requêtes et moyenner les résultats obtenus donnerait un résultat plus précis, pouvant ainsi briser la propriété de confidentialité différentielle), le contrôle d'accès, *etc.*

2.5 Résumé

L'approche proposée dans ce manuscrit se base, comme PeerReview, sur la notion de responsabilité et vise à détecter les tricheries des utilisateurs en vérifiant leurs actions. Le modèle considéré est celui d'utilisateur rationnel, tel qu'introduit par la théorie des jeux. Cependant, il est difficile, dans le cas du sondage dans un réseau social par exemple, d'exprimer leur fonction d'intérêt comme une combinaison de la réputation et de l'impact sur le résultat, on utilise par conséquent un ensemble d'objectifs ordonnés par priorité. La modélisation de la puissance de calcul des utilisateurs est centrée sur le besoin d'extensibilité, qui caractérise la capacité d'un système à gérer un grand nombre d'entités. Pour cette raison, la complexité en temps et en message est également prise en compte. Dans ce manuscrit, la confidentialité est définie de manière similaire au k-anonymat mais de manière probabiliste et elle est assurée à l'aide de mécanismes de partage de secret par homomorphisme non-cryptographique dérivant de la théorie de l'information. De manière générale, l'approche proposée dans ce manuscrit explore les alternatives à la cryptographie en exploitant à la place les caractéristiques du contexte applicatif, par exemple l'aspect aléatoire des protocoles épidémiques et l'aspect social des utilisateurs.

CHAPITRE 3

SYSTÈMES PAIR-À-PAIR

Cette partie est consacrée à l'étude d'un type de tricheurs spécifiques aux systèmes collaboratifs ou pair-à-pair de dissémination de contenus volumineux : les profiteurs (*freeriders*) qui tentent de minimiser leur contribution au système tout en maximisant les bénéfices qu'ils en retirent. Ce travail a été mené en collaboration avec Rachid Guerraoui (EPFL) et Maxime Monod (EPFL) et a abouti à la conception de LiFTinG (pour *Lightweight Freerider Tracking in Gossip*) : un protocole léger de détection des profiteurs pour les systèmes de dissémination épidémique. LiFTinG est composé d'un ensemble de vérifications déterministes et statistiques effectuées par les utilisateurs eux-mêmes en recoupant les informations contenues dans leur historique et d'un mécanisme de détection des profiteurs basé sur l'analyse des blâmes émis par les vérifications. Le fait que les utilisateurs se surveillent entre eux combiné au caractère aléatoire de la méthode de sélection des collaborateurs dans les systèmes de dissémination épidémique assurent à LiFTinG une résistance aux coalitions de profiteurs qui tenteraient de se couvrir mutuellement, et ce, sans avoir recours à la cryptographie pour certifier la véracité des historiques.

Ce travail comporte une présentation du protocole LiFTinG ainsi qu'une étude théorique de ses performances permettant de fixer précisément ses paramètres en fonction des performances souhaitées. L'aspect pratique de LiFTinG est démontré par la conception et le déploiement sur PlanetLab d'un système réel de diffusion épidémique de contenus volumineux sécurisé contre les profiteurs à l'aide de LiFTinG, basé sur des travaux antérieurs de Maxime Monod [FGK+09b].

Une version préliminaire de LiFTinG a été présentée à la conférence P2P 2009 (9th IEEE International Conference on Peer to Peer Computing) [GHKM09b] et une version complète apparaît dans les actes de la conférence Middleware 2010 (11th ACM/IFIP/USENIX International Middleware Conference) [GHK+10].

3.1 Introduction

La diffusion de contenus volumineux tels que des flux vidéo (*streaming*) ou des mises à jour logicielles représente aujourd'hui une proportion non négligeable du trafic Internet et constitue un

exemple typique de service coûteux dans sa version centralisée pouvant bénéficier grandement du paradigme pair-à-pair. Dans un système de dissémination pair-à-pair de contenus, les utilisateurs participent à la diffusion en relayant les données qu'ils reçoivent à d'autres utilisateurs, assurant ainsi une diffusion à faible coût : la quantité de bande-passante nécessaire à la dissémination du contenu est répartie parmi les utilisateurs et n'incombe donc pas uniquement à la source. Les protocoles épidémiques basés sur l'infection ou le commérage (*gossip*) ont été utilisés avec succès pour la conception de systèmes de diffusion pair-à-pair pour leur simplicité et leurs performances. Le principe, inspiré de l'épidémiologie, est le suivant : chaque membre du système (la source comme les utilisateurs) sélectionne périodiquement un ou plusieurs utilisateurs choisi(s) aléatoirement et leur transmet des données qu'il a reçues par le passé, à l'image d'un virus se propageant lors de la rencontre entre un individu sain et un individu infecté. Un tel protocole est dit *asymétrique* dans la mesure où les transferts de données sont unidirectionnels et sont initiés de manière unilatérale par l'émetteur. Initialement conçus pour la diffusion d'informations de petite taille (par opposition à un contenu volumineux), les protocoles épidémiques se révèlent très efficaces en termes de rapidité de dissémination, c'est-à-dire de temps nécessaire à ce que tous les utilisateurs aient reçu l'information, et de résistance aux pannes. Ils ne peuvent cependant pas être utilisés directement pour diffuser du contenu volumineux du fait de la forte redondance qu'ils génèrent, un utilisateur recevant plusieurs fois la même information.

Une variante dite à trois phases résout ce problème en ne transmettant que les informations innovatrices pour le récepteur, c'est-à-dire celles que celui-ci n'a pas reçues par le passé. Les utilisateurs proposent les données qu'ils ont reçues à un ensemble d'utilisateurs choisis aléatoirement. Chacun d'entre eux détermine et demande alors en retour celles qui sont innovatrices pour lui et ces données lui sont finalement servies, c'est-à-dire envoyées par celui qui les avaient proposées. Ainsi, la localisation du contenu est disséminée de manière épidémique alors que le contenu lui-même est diffusé à l'aide d'un protocole classique à deux phases identique à celui utilisé dans une architecture client-serveur. Les protocoles à trois phases sont couramment utilisés dans les systèmes de diffusion épidémique de contenus volumineux tels que [FGK+09a, LCM+08, DXL+06] ou, dans une version légèrement modifiée, dans les systèmes dit à mailles [LXQ+08, ZZSY07, VYF06], c'est-à-dire où chaque utilisateur est connecté à un ensemble statique d'utilisateurs avec qui il échange des données.

L'efficacité des protocoles pair-à-pair dépend grandement du bon vouloir des utilisateurs à collaborer, c'est-à-dire à consacrer une partie de leur bande-passante montante à l'envoi de données aux autres utilisateurs. Il s'avère cependant que certains utilisateurs profitent du système (*freeride*) [AH00, HCW05] sans jamais y contribuer. En d'autres termes, ces utilisateurs que l'on appellera des profiteurs, ne relaient pas les données qu'ils reçoivent, économisant ainsi leur bande-passante montante pour l'utiliser à d'autres fins tout en continuant à bénéficier du système, puisque les données leur sont tout de même envoyées par les autres utilisateurs honnêtes ou généreux. La bande-passante montante est en effet une ressource limitée, la plupart des utilisateurs étant connectés à Internet au moyen d'une ligne ADSL, et fortement utilisée par des applications telles que la téléphonie sur IP. Ce genre d'attitudes est courant dans les systèmes déployés sur Internet et accessibles au grand public où les utilisateurs disposent d'un certain anonymat [AH00, HCW05] et entraîne une dégradation significative des performances globales des systèmes collaboratifs [KSTT04]. Il est important de noter ici que si dans les protocoles épidémiques classiques (c'est-à-dire à une phase) un profiteur est équivalent à un utilisateur en panne et que son impact est limité, ce n'est pas le cas dans leur variante à trois phases. En effet, un utilisateur en panne ne contribue certes pas mais il ne consomme pas non plus la bande-passante montante des autres utilisateurs puisqu'il ne répond pas

aux propositions et donc aucun contenu ne lui est envoyé. La bande-passante montante disponible par utilisateur reste donc la même en présence de pannes. En revanche, un profiteur consomme de la bande-passante sans en fournir au système. On peut d'ores et déjà préfigurer de la difficulté à détecter les profiteurs dans les protocoles épidémiques dans le sens où le choix des utilisateurs à qui relayer les données reçues est aléatoire et donc imprévisible.

La tendance à profiter du système peut se manifester de manière collective : une coalition d'utilisateurs tente de minimiser sa contribution, c'est-à-dire la quantité de données envoyées ou la bande-passante montante consacrée à des utilisateurs en dehors de la coalition, tout en maximisant son bénéfice. On peut, par exemple, imaginer un groupe d'utilisateurs relayant les données qu'ils reçoivent uniquement aux autres utilisateurs du groupe à la manière d'un puits. Cet exemple illustre la difficulté du problème de détection des profiteurs dans la mesure où, dans le cas présent, ces derniers relaient effectivement les données. D'autre part, l'éventualité que des utilisateurs puissent collaborer pour servir leur intérêt commun au détriment du système laisse entrevoir les difficultés rencontrées pour concevoir un protocole de surveillance des utilisateurs par les utilisateurs eux-mêmes : des profiteurs qui se couvrent mutuellement pour déjouer la surveillance ou qui ne rapportent pas les écarts au protocole des autres utilisateurs de la coalition.

Une technique largement employée pour lutter contre les profiteurs est le mécanisme incitatif dit du donnant-donnant (TfT pour *Tit-for-Tat*) introduit par BitTorrent, un protocole collaboratif, très populaire, d'échange de fichiers [Coh03], et utilisé par exemple dans FlightPath [LCM+08]. Ce mécanisme consiste à n'autoriser que les échanges bidirectionnels entre les utilisateurs : chaque utilisateur doit envoyer en échange des données reçues la même quantité de données à l'expéditeur. Ainsi, il assure par le biais d'échanges symétriques et équilibrés que la contribution d'un utilisateur est égale à son bénéfice. Bien que fonctionnels et largement utilisés pour l'échange collaboratif de fichiers, les systèmes utilisant TfT sont sujets à de nombreuses attaques [LMSW06, SPCY07, PIA+07] sur une de leurs composantes, indispensable au bon fonctionnement de TfT, l'envoi généreux (*opportunistic unchoking* en anglais), et sont au final bien moins efficaces en termes de qualité de service (fluidité et qualité de l'image dans un système de diffusion de vidéos par exemple) et de capacité à supporter un grand nombre d'utilisateurs que les systèmes asymétriques pour la diffusion de flux vidéo.

En pratique, dans la majorité des systèmes publiés ou déployés, les données sont diffusées de manière asymétrique [FGK+09b, FGK+09a, LXQ+08, ZZSY07, VYF06]. Les utilisateurs sont supposés relayer généreusement les données qu'ils reçoivent sans rien demander en échange. Ainsi, le bénéfice d'un utilisateur n'est pas directement lié à sa contribution mais plutôt à la générosité moyenne des utilisateurs du système. Si une grande proportion d'utilisateurs ne contribue pas, le bénéfice, et donc la qualité de service de chacun, s'en voit dégradé. Une corrélation binaire entre la contribution et le bénéfice d'un utilisateur peut être établie artificiellement de manière coercitive en surveillant les utilisateurs du système et en excluant ceux considérés comme des profiteurs : les utilisateurs ne contribuant pas équitablement au système ne peuvent plus en bénéficier. Dans une telle optique, le comportement d'un profiteur rationnel peut être modélisé par un utilisateur qui minimise sa contribution sous la contrainte de ne pas être détecté par le protocole de surveillance.

Ce travail considère un protocole épidémique générique à trois phases où les données sont diffusées de manière asymétrique et présente dans ce contexte LiFTinG, un protocole léger et décentralisé de détection des profiteurs. À notre connaissance, LiFTinG est le premier protocole permettant de sécuriser les systèmes de diffusion épidémique contre des profiteurs pouvant former des coalitions. LiFTinG se distancie également des autres protocoles proposés jusqu'à maintenant

car il n'utilise pas de cryptographie asymétrique contrairement aux systèmes [LCM+08, LCW+06, CN03,JMB04,NWD03,FCC+03] et qu'il considère un réseau dynamique où les utilisateurs avec qui un utilisateurs échange des données change régulièrement, contrairement aux systèmes considérés dans [NNS+05, KKU08, HJPVR08]. D'autre part, le surcoût en bande-passante causé par LiFTinG reste très faible comparé à celui de la diffusion proprement dite et peut être adapté dynamiquement suivant la générosité moyenne du système : par exemple, la cadence des vérifications et la quantité d'information échangée lors de celles-ci peuvent en effet être augmentées lorsque la qualité de service des utilisateurs est dégradée de sorte à exclure les profiteurs et retrouver des performances acceptables pour les autres utilisateurs.

Le cœur de LiFTinG est composé d'un ensemble de vérifications réparties, c'est-à-dire effectuées par les utilisateurs eux-mêmes. Les vérifications sont déterministes ou statistiques et utilisent l'historique des actions effectuées par les utilisateurs. Chaque utilisateur est chargé de maintenir un tel registre et les informations contenues dans l'historique des utilisateurs sont recoupées pour détecter les utilisateurs déviant du protocole pour profiter du système. Les vérifications déterministes surveillent que les données envoyées à un utilisateur sont effectivement relayées comme spécifié par le protocole (en temps et en heure, au bon nombre d'utilisateurs, etc.). Les vérifications statistiques, quant à elles, surveillent que les valeurs des paramètres contrôlant les parties aléatoires du protocole de diffusion, plus particulièrement le mécanisme de sélection des utilisateurs à qui relayer les données reçues, sont bien conformes aux spécifications.

Étonnamment, la composante aléatoire des protocoles épidémiques qui semblait à première vue être un obstacle à la surveillance et à la détection des profiteurs se révèle être un atout de poids dans la lutte contre les coalitions. En effet, le fait qu'un utilisateur interagisse avec (c'est-à-dire reçoive des données de et envoie des données à) un grand nombre d'utilisateurs choisis aléatoirement et répartis uniformément dans le système limite les possibilités que les membres d'une coalition se couvrent mutuellement : en supposant que la taille des coalitions reste négligeable devant la taille totale du système, l'historique des actions d'un utilisateur obtenu en recoupant les historiques des utilisateurs avec qui il a interagi est globalement digne de confiance et peut être utilisé pour juger sa contribution. La combinaison de ces deux types de vérifications donne de très bons résultats en termes de détection avec un fort taux de détection pour un faible taux de faux positifs (c'est-à-dire que peu d'utilisateurs sont considérés à tort comme des profiteurs).

Pour quantifier l'efficacité de LiFTinG ainsi que pour permettre de le paramétrer précisément en fonction des performances souhaitées, ce travail contient une analyse théorique du mécanisme de détection de LiFTinG illustrée par des simulations numériques pour des systèmes de plusieurs dizaines de milliers d'utilisateurs. L'aspect pratique de LiFTinG est démontré par un déploiement réel d'un système de diffusion épidémique d'un flux vidéo de débit 674kbps sécurisé par LiFTinG sur 300 ordinateurs de la plate-forme expérimentale PlanetLab. Dans ce contexte expérimental, le surcoût en termes de bande-passante induit par LiFTinG reste inférieur à 8%. En présence de profiteurs diminuant leur contribution de 30%, LiFTinG réussit à détecter 86% d'entre eux après 30 secondes et la proportion d'utilisateurs exclus à tort à ce point est de 12% et concerne majoritairement des utilisateurs ayant de très faibles capacités en termes de bande-passante qui peuvent donc être considérés comme des profiteurs (PlanetLab étant utilisé par les chercheurs du monde entier, il est fréquent que certains ordinateurs soient surchargés et ne soient pas en mesure de supporter un flux vidéo à haut débit).

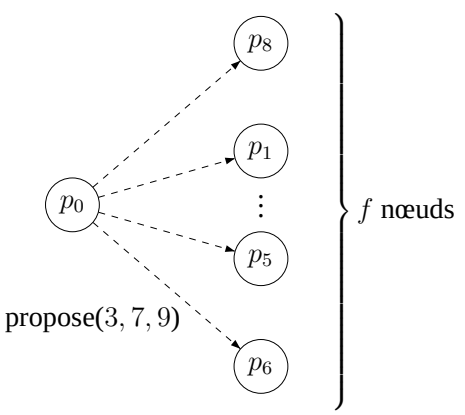
3.2 Protocole épidémique

Dans le cadre de ce travail, on considère un système composé de n utilisateurs pouvant communiquer via des canaux avec pertes (une connexion UPD sur Internet par exemple). Les ordinateurs des utilisateurs du système constituent ainsi les nœuds d'un réseau de communication. On supposera que n'importe quelle paire de nœuds du système est capable d'établir une connexion et de transférer des données par ce biais. Cela suppose que le pare-feu des ordinateurs est configuré pour autoriser les connexions et que la translation d'adresse est correctement effectuée par le routeur pour les ordinateurs appartenant à un réseau interne. Dans le cas d'un système déployé pour le grand public cette hypothèse est très souvent vérifiée, la plupart des utilisateurs étant connectés à Internet directement par un modem ou derrière un routeur ADSL qui autorise la redirection manuelle et automatique de port (via l'UPnP pour *Universal Plug'n Play*), rendant ainsi accessible l'ordinateur depuis Internet. On notera également que des techniques telles que la perforation de routeurs (*hole punching*) ou des mécanismes utilisant un nœud tiers [KPQS09] permettent d'établir une connexion entre deux nœuds placés derrière un routeur.

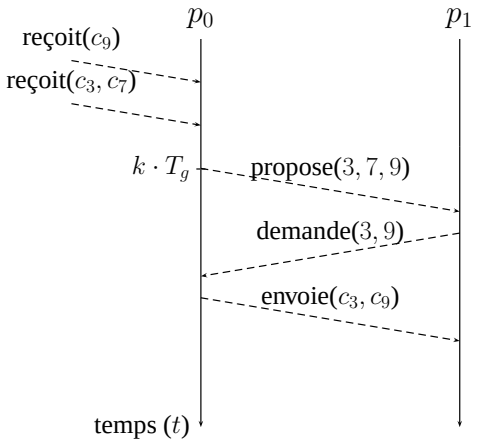
On supposera également que les nœuds peuvent choisir un ensemble aléatoire de nœuds parmi tous les nœuds du système à l'aide d'une méthode d'échantillonnage uniforme (RPS pour *random peer sampling*) [JGKvS04]. Il existe à ce jour plusieurs techniques d'échantillonnage, la plus simple consistant à maintenir à jour sur chaque nœud la liste exhaustive des autres nœuds du système. Pour des raisons d'extensibilité mais surtout à cause de la forte dynamique des systèmes pair-à-pair (*churn*) cette solution est souvent évitée et avantageusement remplacée par des algorithmes répartis utilisant des listes partielles, appelées *vues*, d'utilisateurs du système qui sont échangées entre les nœuds par commérage [KS04, VGvS05, MLMKG06, JVG$^+$07, EGH$^+$03]. On notera que ces techniques peuvent être rendues résistantes aux attaques byzantines grâce à des protocoles tels que Brahms [BGK$^+$09]. Un profiteur aurait en effet intérêt à attaquer le protocole d'échantillonnage réparti afin d'être choisi plus fréquemment que les autres nœuds lors de la sélection des nœuds à qui des données sont proposées et ainsi augmenter son bénéfice. Enfin, on supposera que les nœuds connaissent la taille n du système. À nouveau, cette information peut être calculée de manière efficace en utilisant des protocoles de comptage par commérage [KPG$^+$05] ou par marche aléatoire [MLMKG06].

Le contenu à disséminer est un flux vidéo diffusé par une unique source à l'ensemble des n nœuds du système à l'aide d'un protocole épidémique à trois phases tel que celui utilisé dans [FGK$^+$09b, DXL$^+$06]. Le flux est découpé en blocs de taille constante identifiés de manière unique par des entiers. On supposera que la taille (en octet) d'un bloc est très grande devant la taille d'un identifiant. En bref, chaque nœud propose à intervalles réguliers les blocs qu'il a reçus au cours du dernier intervalle de temps à un nombre fixe de nœuds choisis aléatoirement de manière uniforme. À la réception d'une proposition de blocs, un nœud demande les blocs dont il a besoin à l'expéditeur de la proposition qui les lui envoie alors. Tous les messages échangés le sont via des connexions UDP. Les trois phases sont décrites formellement et en détail dans les paragraphes suivants et illustrées sur la figure 3.1.

Phase de proposition Chaque nœud choisit périodiquement, c'est-à-dire toutes les T_g unités de temps, au moyen d'un algorithme d'échantillonnage *uniforme* un ensemble de f nœuds du système et leur propose l'ensemble $\mathcal{P}$ des identifiants des blocs qu'il a reçus lors de la dernière période. Le paramètre f, appelé arité sortante (*fan-out*), est le même pour tous les nœuds du système et gardé



(a) Proposition épidémique : p_0 choisit f nœuds aléatoirement à qui il propose des blocs (ici les blocs numéro 3,7 et 9).



(b) Trois phases : p_0 propose les blocs numéro 3,7 et 9 qu'il a reçus durant la période précédente aux nœuds choisis. Ici, p_1 demande les blocs 3 et 9 à p_0 qui les lui envoie.

FIGURE 3.1 : Protocole de dissémination épidémique à trois phases.

constant dans le temps. On remarquera qu'un bloc reçu par un utilisateur n'est proposé qu'une fois par ce dernier, à savoir pendant la période suivant sa réception. Il est prouvé que ce type de méthode de diffusion épidémique, appelé contamine-et-meurt (*infect-and-die*), assure que chaque nœud du réseau reçoit tous les blocs avec forte probabilité pour une arité sortante f supérieure à $\ln n$ [KMG03].

Phase de demande À la réception d'une proposition contenant un ensemble $\mathcal{P}$ d'identifiants de blocs, un nœud détermine ceux dont il a besoin, c'est-à-dire qu'il n'a pas déjà reçus et qui sont ou seront utiles à la lecture du flux vidéo. En effet, supposons pour simplifier que chaque bloc de données code une séquence du flux, disons une seconde de vidéo, alors les blocs d'identifiant inférieur à soixante ne sont plus d'aucune utilité à un utilisateur ayant déjà visionné soixante secondes de vidéo. On notera que si un bloc n'est pas disponible au moment où la séquence correspondante doit être lue, la lecture passe au bloc suivant entrainant une discontinuité dans la vidéo. Par conséquent un bloc peut être inutile pour un utilisateur sans pour autant avoir déjà été reçu par ce dernier. On note $\mathcal{R}$ l'ensemble des identifiants des blocs demandés.

Phase d'envoi Lorsqu'un nœud reçoit une demande répondant à une proposition qu'il a envoyée, il envoie les blocs demandés. On note $\mathcal{S}$ l'ensemble des blocs envoyés. Seule cette phase est réellement consommatrice de bande-passante dans la mesure où la taille d'un ensemble d'identifiants est négligeable devant la taille d'un bloc de données. Pour se protéger des demandes opportunistes et intempestives les nœuds ignorent les demandes ne répondant pas à une proposition qu'ils ont émise et n'envoient que des blocs effectivement proposés, c'est-à-dire les blocs dont les identifiants sont dans $\mathcal{P} \cap \mathcal{R}$.

3.3 Profiteurs

Le système est composé de deux types de nœuds : les nœuds honnêtes et les profiteurs. Les nœuds honnêtes suivent scrupuleusement le protocole, y compris les vérifications de LiFTinG. En revanche les profiteurs s'autorisent à dévier du protocole pour diminuer leur contribution, c'est-à-dire le nombre de blocs qu'ils envoient, et augmenter leur bénéfice, c'est-à-dire le nombre de blocs qu'ils reçoivent. Cela inclut les attaques contre les procédures de vérifications dans le but d'éviter l'exclusion qui réduirait leur bénéfice à zéro. Il est important de noter ici que si accuser à tort des nœuds ne diminue pas les risques pour un profiteur d'être exclu, alors il n'a pas intérêt à le faire. D'autre part, faire en sorte que des nœuds honnêtes soient exclus diminue la générosité moyenne du système et donc le bénéfice de chacun. Ce phénomène est connu sous le nom de tragédie des biens communs (*tragedy of the commons*). En conclusion, on supposera que les profiteurs n'émettent pas de blâmes injustifiés à l'encontre des nœuds honnêtes.

De manière générale un nœud peut dévier d'un protocole de trois manières : (1) en biaisant le choix des nœuds avec qui il interagit, (2) en envoyant plus ou moins de messages qu'il n'est censé le faire et (3) en modifiant le contenu de ces messages. Dans la suite, on liste les déviations possibles dans chacune des trois phases du protocole épidémique présenté dans la section précédente et on identifie celles pouvant servir l'intérêt individuel ou commun des profiteurs. Compte tenu du modèle de profiteurs précédemment énoncé, seules ces déviations sont implémentées par ces derniers et doivent donc être contrées. Les déviations qui nécessitent l'aide d'un tiers appartenant à une coalition ou qui servent l'intérêt commun d'une coalition sont marquées d'un « $\star$ ».

3.3.1 Phase de proposition

Au cours de la première phase du protocole, un profiteur peut (1) choisir des nœuds de manière non uniforme (dans un ensemble restreint par exemple), (2) choisir moins de f nœuds, (3) proposer moins de blocs qu'il ne devrait et (4) diminuer la fréquence à laquelle il envoie des propositions.

1. **Biais de l'échantillonnage ($\star$)** Dans le cas d'une coalition de profiteurs, un profiteur peut favoriser les membres de sa coalition lors de la phase d'échantillonnage de façon à augmenter leur bénéfice commun, brisant ainsi l'uniformité de l'échantillonnage aléatoire.
2. **Diminution de l'arité sortante** En proposant des blocs à seulement $\hat{f}$ nœuds par période avec $\hat{f} < f$, un nœud diminue trivialement le nombre de demandes qu'il reçoit et donc le nombre de blocs à envoyer. Par conséquent, sa contribution au système est diminuée.
3. **Proposition incomplète** Une proposition est complète si elle contient l'intégralité des blocs reçus par le nœud lors de la dernière période. En ne proposant qu'un sous-ensemble de ces blocs, un nœud diminue potentiellement le nombre de blocs demandés et donc sa contribution. On notera cependant que contrairement aux systèmes donnant-donnant, un nœud n'a pas intérêt à proposer des blocs qu'il n'a pas dans le but de demander plus de blocs en retour, les échanges étant ici asymétriques.
4. **Augmentation de la période** En augmentant sa période de proposition un nœud propose des blocs moins souvent mais chaque proposition contient plus d'identifiants de blocs, le nombre de total de blocs proposés restant le même. Cependant l'âge moyen des blocs proposés (le temps qui sépare leur injection dans le système par la source de l'instant présent) est augmenté et donc la probabilité que ces blocs soient utiles est diminuée. En effet, au fur et à mesure que

le temps passe, la probabilité qu'un nœud ait reçu un bloc donné augmente et l'avancement des utilisateurs dans la lecture du flux également, diminuant ainsi la probabilité que ce bloc soit utile. Par conséquent, cette déviation diminue potentiellement le nombre de blocs qu'un nœud doit servir.

3.3.2 Phase de demande

Les nœuds sont supposés ne demander que des blocs qui leur ont été proposés. Un profiteur pourrait augmenter son bénéfice en demandant des blocs de manière opportuniste (éventuellement à des nœuds qui ne lui en ont pas proposé). Cependant, le protocole lui-même empêche cette déviation en ignorant automatiquement de telles demandes.

3.3.3 Phase d'envoi

Lors de la phase d'envoi, un nœud peut n'envoyer qu'un sous-ensemble de ce qui lui a été demandé ou envoyer un contenu incorrect, c'est-à-dire autre chose que le contenu de ce bloc tel qu'il a été émis par la source. La première déviation diminue la contribution du nœud. Pour les raisons évoquées précédemment, la seconde déviation, qui augmenterait le bénéfice d'un nœud dans un système donnant-donnant, n'a pas d'effet ici.

3.3.4 Résumé

L'analyse du protocole de dissémination épidémique a permis d'identifier les déviations qui augmentent le bénéfice d'un nœud ou diminuent sa contribution. Compte tenu du modèle de tricheurs considéré, à savoir des profiteurs, seules ces déviations sont considérées. On note que ces déviations partagent des points communs qui dictent les angles sous lesquels les procédures de vérification de LiFTinG sont conçues.

Le premier point est la validité quantitative du protocole, c'est-à-dire que chaque nœud contacte effectivement f nœuds par période de T_g unités de temps. Une fois ce premier aspect vérifié il reste à assurer la causalité, qui constitue la partie déterministe du protocole, et la validité statistique. Ces deux propriétés signifient respectivement que les blocs reçus sont effectivement relayés et que les nœuds auxquels ils le sont sont bien choisis uniformément dans le système. Dans la mesure où l'approche proposée dans ce travail est composée de vérifications effectuées par les nœuds avec qui un nœud interagit, il est important de noter que l'uniformité de l'échantillonnage a pour but non seulement l'équité de la diffusion mais également la diversité des vérificateurs d'un nœud.

3.4 LiFTinG : détection légère des profiteurs

LiFTinG est un protocole de détection des profiteurs encourageant, de manière coercitive à l'aide de mécanismes de vérification répartie, les utilisateurs à contribuer équitablement au système. LiFTinG est composé de deux types de vérifications : les vérifications *instantanées* et les vérifications *a posteriori*. Certaines vérifications nécessitent de recouper des informations venant de différents nœuds ayant interagi avec le nœud sujet à ces vérifications. Elles sont appelées vérifications par recoupement. Ces dernières sont la cause du surcoût de bande-passante engendré par LiFTinG car elles nécessitent des échanges d'informations entre les nœuds en plus des trois phases nécessaires

aux échanges de blocs. La fréquence à laquelle les vérifications par recoupement sont déclenchées est contrôlée par un paramètre p_{cc}. On reviendra plus en détail sur ce paramètre par la suite.

Les vérifications dites instantanées sont intégrées au protocole de dissémination et vérifient à la volée que les nœuds suivent effectivement le protocole. Elles assurent la validité quantitative de leurs actions ainsi que la causalité telles qu'elles ont été décrites dans la section précédente. Parmi elles, certaines nécessitent des échanges d'informations entre les nœuds : le recoupement instantané.

Les vérifications *a posteriori* sont déclenchées de manière sporadique par les nœuds sur des nœuds choisis aléatoirement et uniformément. Ces vérifications ont pour but de contrôler le comportement des nœuds sur le long terme. Elles nécessitent que chaque nœud conserve un historique de ses actions et interactions avec les autres nœuds du système. En pratique il ne s'agit que d'un résumé des événements (contenant par exemple l'identifiant d'un nœud et les identifiants des blocs qu'il a envoyés mais pas le contenu des blocs) qui se sont déroulés pendant les h dernières unités de temps, l'équivalent de $n_h = h/T_g$ périodes. Lors d'une vérification *a posteriori*, l'historique du nœud lui est demandé et est inspecté afin qu'on s'assure en particulier de l'uniformité de la partie aléatoire du protocole mais également des aspects quantitatifs, de la causalité et de la période de proposition. Pour le premier point, il s'agit de vérifications statistiques. La véracité de l'historique est vérifiée en recoupant les informations qu'il contient avec l'historique des nœuds apparaissant dans ce dernier.

Les vérifications effectuées par les nœuds peuvent conduire à l'émission de blâmes ou d'ordres d'expulsion directe suivant la gravité de la faute détectée. LiFTinG utilise une architecture complémentaire pour gérer la surveillance des nœuds, la collecte des blâmes et l'exclusion. Le tableau 3.1 récapitule les différentes déviations listées dans la section précédente, leur type et les vérifications permettant de les détecter.

Déviation	**Type**	**Détection**
échantillonnage biaisé ($\star$)	statistique	vérification statistique, recoupement *a posteriori*
diminution de l'arité sortante	quantitative	recoupement instantané
proposition incomplète	causalité	recoupement instantané
augmentation de la période	quantitative	vérification statistique, recoupement *a posteriori*
envoi incomplet	quantitative	vérification instantanée

TABLEAU 3.1 : Résumé des déviations et vérifications associées.

La suite de cette section est organisée de la manière suivante : la sous-section 3.4.1 présente l'architecture de détection de LiFTinG. Les sous-sections 3.4.2, 3.4.3 et 3.4.4 décrivent respectivement les vérifications instantanées, les attaques possibles contre ces vérifications et les vérifications *a posteriori* (qui contrecarrent les attaques mentionnées précédemment). Le surcoût de LiFTinG est analysé de manière théorique dans la sous-section 3.4.5.

3.4.1 Architecture de blâme

Pour détecter les profiteurs, LiFTinG se base sur des scores numériques attribués aux nœuds du système et déterminés par les résultats des procédures de vérification. Lorsqu'au cours d'une vérification, un nœud est suspecté d'avoir dévié du protocole, le nœud effectuant la vérification émet un blâme, contenant une valeur numérique négative, à son encontre. Le score d'un nœud, initialisé

à zéro est la somme des blâmes émis à son encontre depuis son entrée dans le système. Pour déterminer si un nœud doit être exclu ou non, LiFTinG se base sur plusieurs paramètres tels que le score du nœud et le temps passé dans le système. Un tel fonctionnement pose un certain nombre de questions et entraîne certaines contraintes dans la conception du protocole. Par exemple, comment collecter les blâmes émis à l'encontre d'un nœud donné ? Ou encore comment exclure un nœud du système ? D'autre part pour assurer que le score d'un nœud résultant de l'agrégation de blâmes émis par différentes vérifications soit pertinent, il convient que les valeurs des blâmes soient homogènes. C'est-à-dire qu'elles correspondent à une quantité réelle et qu'elles aient par conséquent la même *dimension*. Par analogie à la physique, cela a peu, voire pas, de sens de sommer un poids et un volume. De même, lorsqu'on additionne deux volumes, il faut vérifier qu'ils ont la même unité. Enfin, pour empêcher que les profiteurs ne blâment à tort les nœuds honnêtes, l'algorithme de décision d'exclusion doit traiter les nœuds de manière indépendante et non dans leur globalité pour éviter que les profiteurs ne puissent échapper à l'exclusion en accusant d'autres nœuds. Par analogie à nouveau, il n'y a aucun intérêt à accuser de tricherie, à tort, un candidat à un examen alors que cela augmente les chances d'être accepté dans un concours à *numerus clausus*. En bref, les solutions apportées à ces problèmes sont les suivantes : l'architecture de blâme utilise le système existant de surveillance AVMON [MG09] ; l'exclusion est effectuée à l'aide de messages de révocation propagés par commérage ; les blâmes émis par les vérifications sont homogènes et leur unité est un nombre d'envois (c'est-à-dire le nombre d'envois économisés par le profiteur) ; l'algorithme de décision d'exclusion se base sur la comparaison du score harmonisé des nœuds à un seuil fixe (à la manière d'un examen). Ces points sont détaillés par la suite. La figure 3.2 donne un aperçu de l'architecture globale de LiFTinG.

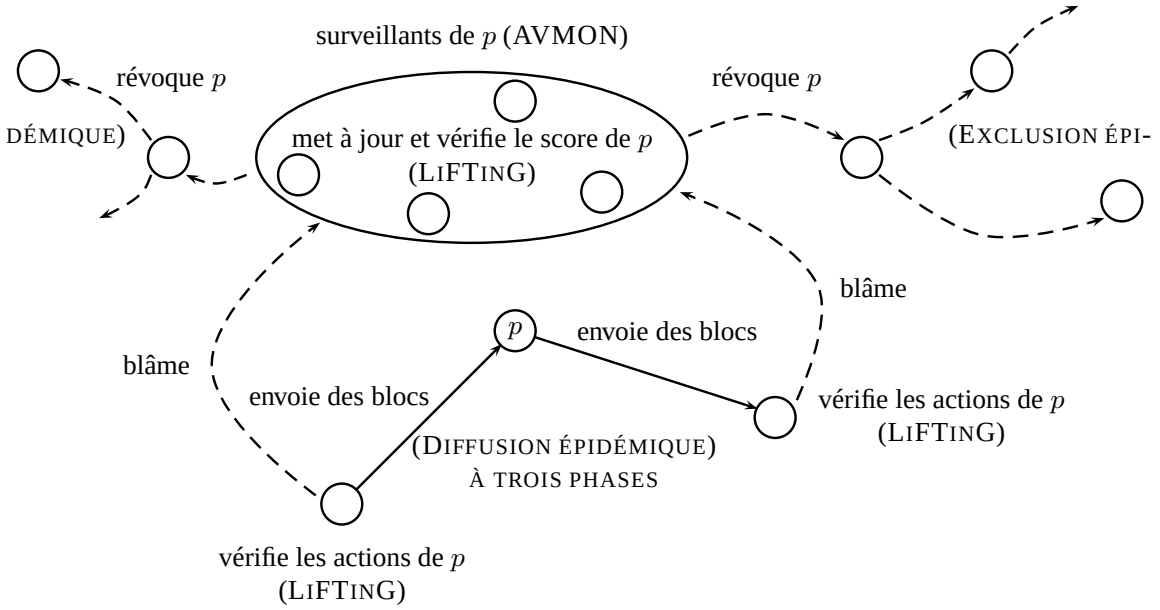


FIGURE 3.2 : Schéma de l'architecture globale de LiFTinG.

L'architecture de blâme de LiFTinG repose sur le système de surveillance AVMON qui assigne à chaque nœud des surveillants choisis aléatoirement parmi les autres nœuds du système : les nœuds

se surveillent mutuellement. Les blâmes émis à l'encontre d'un nœud sont envoyés à chacun de ses surveillants qui mettent alors à jour son score et décident individuellement s'il doit être exclu. Il suffit qu'un surveillant d'un nœud décide de son exclusion pour que ce nœud soit effectivement exclu. Dans AVMON, le nombre de surveillants d'un nœud est fixé à une constante M. Un nœud est surveillé par les mêmes nœuds durant l'intégralité du temps qu'il passe dans le système assurant ainsi un suivi efficace. Cet aspect est particulièrement important dans le contexte de ce travail, à savoir un environnement pair-à-pair dynamique sujet aux connexions/déconnexions intempestives des nœuds, dans la mesure où il permet d'obtenir un score reflétant l'attitude sur le long terme du nœud surveillé. Le fait que le nombre de surveillants d'un nœud soit constant rend le protocole capable de gérer un grand nombre de nœuds. En effet, un nœud est surveillé par M nœuds ce qui signifie que chaque nœud surveille en moyenne M autres nœuds. Par conséquent, la charge de travail due à la surveillance n'augmente pas avec la taille du système. L'aspect aléatoire du choix des surveillants permet d'éviter la collusion entre un nœud et ses surveillants, la probabilité que tous les surveillants d'un nœud appartiennent à la même coalition de profiteurs que lui décroissant exponentiellement (c'est-à-dire comme $(n/m)^M$) et pouvant donc être rendue arbitrairement petite pour des valeurs raisonnables de M. La relation surveillant-surveillé n'est pas réellement aléatoire mais établie à l'aide d'une fonction de hachage appliquée aux identifiants des nœuds, qui eux sont aléatoires et ne peuvent donc pas être choisis. Par exemple, utiliser l'adresse IP des nœuds (qui n'est pas choisie par la majorité des utilisateurs grand public, cette adresse étant attribuée par le fournisseur d'accès à Internet) se révèle suffisant pour lutter contre la collusion entre surveillants et surveillés dans AVMON. La liste des surveillants d'un nœud est diffusée aux autres nœuds du système en se greffant sur les messages utilisés par le protocole d'échantillonnage (les vues partielles par exemple). Cette information peut être vérifiée localement par les nœuds eux-mêmes en calculant une fonction de hachage sans nécessiter de communication. Cela permet ainsi d'éviter que la liste des surveillants d'un nœud soit altérée dans le but que les messages de blâmes à son encontre ne soient pas pris en compte.

L'exclusion des nœuds est implémentée par commérage. Pour initier l'exclusion d'un nœud, un surveillant envoie un message de révocation à f nœuds. Ces derniers retirent le nœud visé de la liste des destinataires potentiels (qu'il s'agisse de la liste complète des nœuds du système ou une vue partielle telle qu'utilisée dans CYCLON) et transmettent à leur tour le message de révocation à f autres nœuds et ainsi de suite.

3.4.2 Vérifications instantanées

LiFTinG utilise deux mécanismes de vérification instantanée. Le premier a pour but de vérifier que tous les blocs demandés sont effectivement reçus. Ce mécanisme peut être exécuté localement par les nœuds sans nécessiter de communiquer avec d'autres nœuds, il est donc toujours exécuté. Si certains blocs demandés ne sont pas reçus, un blâme d'une valeur de $f/|\mathcal{R}|$ par bloc manquant est émis à l'encontre de l'expéditeur (où $\mathcal{R}$ est l'ensemble des blocs demandés).

Le second mécanisme vérifie la bonne propagation des blocs envoyés. Plus précisément, lorsqu'un nœud envoie des blocs à un autre nœud il s'assure ensuite que ce dernier les propose effectivement à f nœuds dans la prochaine période, c'est-à-dire au plus tard $2T_g$ unités de temps après l'envoi. On notera qu'il suffit de s'assurer ici que les blocs reçus sont effectivement proposés, le mécanisme présenté dans le paragraphe précédent permettant de vérifier que les blocs proposés et demandés sont effectivement envoyés. Ce mécanisme nécessite de communiquer avec le nœud à

qui les blocs ont été envoyés mais également avec les nœuds à qui ce dernier propose ces blocs et recoupe les informations obtenues. Plus précisément, le mécanisme de recoupement instantané fonctionne de la manière suivante : un nœud p_1 ayant reçu un bloc c_i du nœud p_0 confirme à p_0 pendant la prochaine période que c_i est effectivement relayé en lui envoyant la liste des f nœuds à qui il a proposé c_i. Avec une probabilité p_{cc} (c'est ici que le paramètre de gestion du surcoût de bande-passante entre en jeu), p_0 demande confirmation à chacun des nœuds de la liste que p_1 leur a effectivement proposé c_i. Les nœuds interrogés confirment ou infirment la réception de la proposition directement à p_0. La figure 3.3 représente l'enchaînement de messages composant le mécanisme de recoupement instantané pour une arité de deux, dans un souci de lisibilité. Le barème des blâmes émis par ce mécanisme est le suivant : (1) un blâme de 1 est envoyé pour chaque identifiant de nœuds manquant dans la liste et donc un blâme de f si la liste n'est pas envoyée ; (2) un blâme de 1 par non-réponse ou réponse négative des nœuds interrogés. On notera qu'un nœud envoie une réponse négative dès lors qu'un bloc devant être proposé ne l'a pas été. Par conséquent un profiteur décidant de ne pas proposer deux des blocs qu'il a reçus a tout intérêt à choisir deux blocs envoyés par le même nœud car dans ce cas, ses propositions sont incomplètes du point de vue d'un seul des nœuds faisant la vérification.

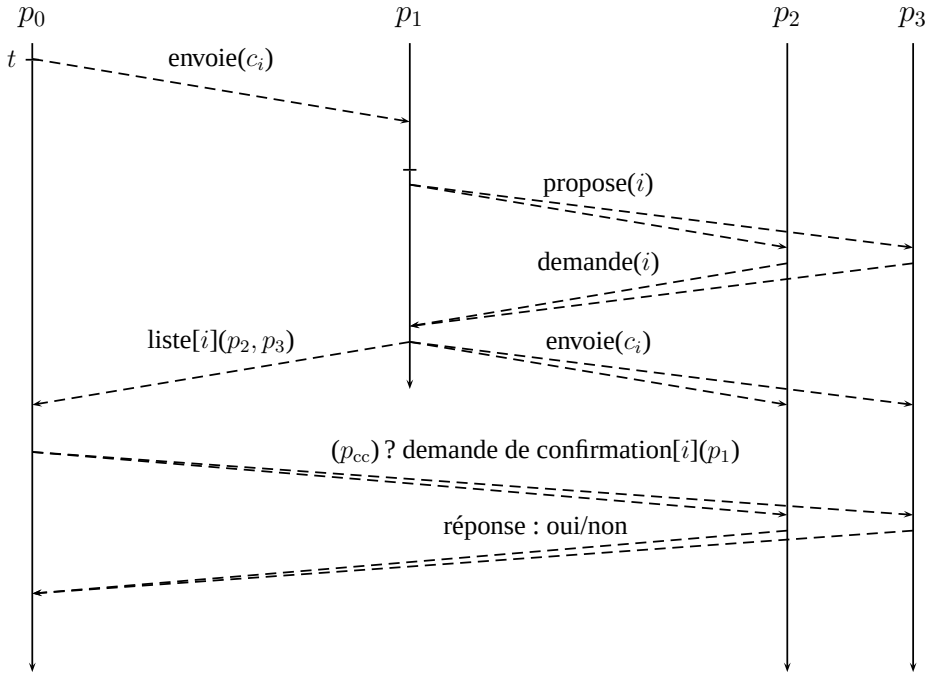


FIGURE 3.3 : Mécanisme de recoupement instantané (pour une arité sortante $f = 2$) : p_0 envoie un bloc c_i à p_1. Ce dernier lui envoie alors, durant la prochaine période, la liste des f nœuds à qui il a proposé c_i. p_0 demande ensuite confirmation à ces derniers.

Pour limiter le surcoût de bande-passante qu'engendrent les vérifications par recoupement, les messages échangés par celles-ci (liste, demande de confirmation et confirmation) le sont via UDP. Ces messages étant de petite taille, le coût de l'établissement (en temps et nombre de messages) d'une connexion TCP est significatif. Le tableau 3.2 résume les valeurs des blâmes pour les diffé-

rentes vérifications instantanées. On notera que dans le premier cas, le blâme indiqué est envoyé par chacun des nœuds ayant envoyé des blocs durant la période précédente, car ils recevront chacun $f - \hat{f}$ réponses négatives. Dans le second cas, seuls les nœuds dont au moins un des blocs qu'ils ont envoyés n'a pas été proposé émettent un blâme.

Déviation	**Valeur du blâme**
diminution de l'arité sortante ($\hat{f} < f$)	$f - \hat{f}$
proposition incomplète	1 (par proposition incomplète)
envoi incomplet ($\|\mathcal{S}\| < \|\mathcal{R}\|$)	$f \cdot (\|\mathcal{R}\| - \|\mathcal{S}\|) / \|\mathcal{R}\|$

TABLEAU 3.2 : Résumé des déviations et des blâmes associés aux vérifications correspondantes.

Comme expliqué dans la section précédente, les blâmes émis par les mécanismes de vérification à l'encontre d'un nœud sont ajoutés à son score et doivent, pour cette raison, être homogènes et de même unité. En d'autres termes, ils doivent quantifier un aspect commun aux différentes déviations et avoir la même valeur pour deux déviations équivalentes. L'unité choisie dans le barème de blâmes de LiFTinG est la proposition (les blâmes correspondent à des propositions non envoyées, incomplètes ou non honorées) et deux déviations donnant le même gain en termes de bande-passante économisée conduisent à la même valeur de blâme.

Pour prouver cette propriété, essentielle au bon fonctionnement du mécanisme de blâme, on considère un profiteur p ayant reçu c blocs durant une période et qui tente de diminuer sa contribution d'un facteur δ ($0 < \delta < 1$). Pour y parvenir, p dispose de trois alternatives : (1) proposer les c blocs reçus à seulement $\hat{f} = (1-\delta) \cdot f$ nœuds, (2) proposer une proportion $(1-\delta)$ des c blocs reçus à f nœuds, ou enfin (3) proposer l'intégralité des c blocs à f nœuds, mais n'envoyer effectivement qu'une proportion $(1 - \delta)$ des $|\mathcal{R}|$ blocs demandés. Dans un souci de simplicité, on supposera que les quantités $\hat{f}, c \cdot \delta$, c/f et $\delta \cdot |R|$ sont toutes des entiers.

Le nombre de nœuds effectuant des vérifications directes sans recoupement est égal au nombre de nœuds qui ont reçu une proposition de p, c'est-à-dire f en temps normal. Le nombre de nœuds effectuant des vérifications directes avec recoupement est égal au nombre de nœuds ayant envoyé des blocs à p au cours de la période précédente (appelé arité entrante et noté f') en considérant qu'une proposition entraîne toujours une demande. En moyenne $f' \sim f$ car un total de $n \cdot f$ propositions sont émises pendant une période, réparties sur un total de n nœuds [GKM01]. Le nombre moyen de blocs servis par chacun de ces nœuds est donc c/f. On détermine maintenant, pour chacune des trois déviations, le blâme appliqué à p par le mécanisme de vérification correspondant.

1. **Diminution de l'arité sortante (vérification instantanée avec recoupement)** Si p propose les c blocs à seulement $\hat{f}$ nœuds, il est blâmé de 1 par chacun des f' nœuds qui effectuent cette vérification, pour chacun des $f - \hat{f}$ nœuds à qui il n'a pas envoyé de proposition. Cela se traduit par un blâme total de $f' \cdot (f - \hat{f}) = f' \cdot \delta \cdot f \simeq \delta f^2$ en moyenne.
2. **Proposition partielle (vérification instantanée avec recoupement)** Si p ne propose que $(1 - \delta) \cdot c$ blocs parmi les c qu'il a reçus, il est blâmé par chacun des nœuds effectuant cette vérification dont au moins un des blocs qu'il a servi n'a pas été proposé par p. Pour ces derniers, les f propositions émises par p sont incomplètes et ils blâment p de 1 pour chacune d'elles. Comme expliqué précédemment, un profiteur a tout intérêt à minimiser le nombre de ces nœuds et donc à ne pas proposer les blocs envoyés par $\delta \cdot f'$ nœuds conduisant à un blâme total de $\delta \cdot f' \cdot f \simeq \delta \cdot f^2$ en moyenne.

3. **Envoi partiel (vérification instantanée sans recoupement)** Si p n'envoie que $(1-\delta)\cdot|\mathcal{R}|$ blocs parmi les $|\mathcal{R}|$ demandés, il est blâmé de $f/|\mathcal{R}|$ pour chacun des $\delta\cdot|\mathcal{R}|$ blocs manquant par chacun des f nœuds à qui il a envoyé une proposition et à qui il devait donc envoyer des blocs. Soit un blâme total de $f\cdot(f/|\mathcal{R}|)\cdot\delta\cdot|\mathcal{R}| = \delta\cdot f^2$.

En conclusion les valeurs des blâmes émis par les mécanismes de vérification instantanée sont bien de même unité et reflètent le gain des profiteurs (égal ici au degré de profit δ). Ils peuvent donc être sommés pour donner un score cohérent et concret utilisé pour décider de l'exclusion des nœuds.

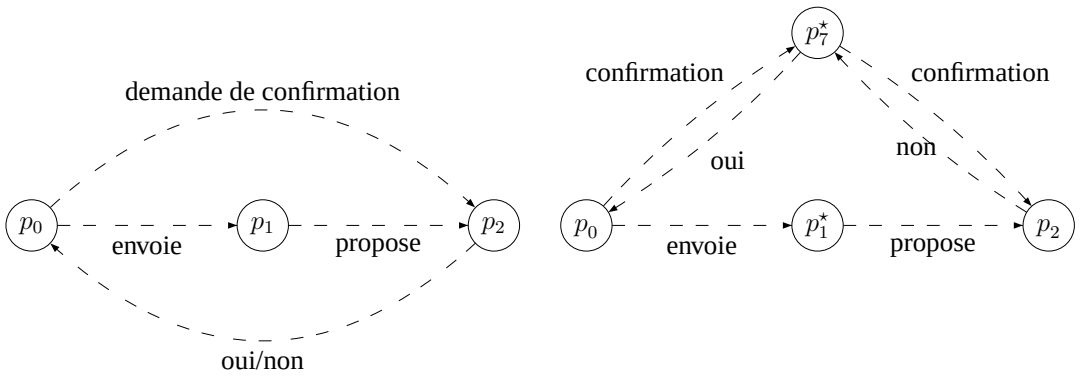
3.4.3 Attaques contre les vérifications instantanées ($\star$)

Dans le cas où les profiteurs forment une coalition, ils peuvent s'entraider pour déjouer les vérifications présentées dans la sous-section précédente. Pour les vérifications instantanées sans recoupement, un profiteur n'émet pas de blâmes à l'encontre d'un profiteur de la même coalition si ce dernier n'envoie pas des blocs proposés, ce qui ne constitue pas un réel problème en soit, le but de LiFTinG étant de protéger l'intérêt des nœuds honnêtes. Dans le cas des vérifications instantanées avec recoupement cependant, un profiteur peut, avec l'aide d'un tiers, se dispenser de proposer les blocs reçus sans pour autant être blâmé. Considérons le cas décrit sur la figure 3.4a et supposons que p_1 est un profiteur. Si p_0 fait partie de la même coalition, il n'émet pas de blâme à l'encontre de p_1 quelle que soit la réponse de p_2. Si p_2 fait partie de la même coalition que p_1, il répondra toujours positivement à la demande de confirmation de p_0 que p_1 lui ait effectivement proposé les blocs ou pas. Pour les mêmes raisons qu'évoquées précédemment, cela ne constitue pas un réel problème. Cependant, même dans le cas où ni p_0 ni p_2 ne font partie d'une coalition incluant p_1, ce dernier peut tout de même déjouer la vérification à l'aide d'un tiers, p_7 par exemple, comme illustré sur la figure 3.4b. Cette technique, connue sous le nom de l'attaque de l'homme au milieu (*man-in-the-middle attack*) consiste à utiliser un tiers qui s'intercale entre deux nœuds échangeant des messages et en modifie le contenu. En effet, les nœuds p_0 et p_2 étant mis en relation par p_1 lui-même, ce dernier peut intercaler un nœud appartenant à sa coalition en envoyant l'identifiant de p_7 (au lieu de celui de p_2) à p_1. p_7 répond alors positivement à la demande de confirmation envoyée par p_0 et envoie une demande de confirmation à p_2 sans tenir compte de la réponse de ce dernier. Ainsi, ni p_0, ni p_2 ne peuvent se rendre compte de l'attaque et p_1 peut donc envoyer une proposition incomplète sans être détecté ni blâmé. Les vérifications *a posteriori* présentées dans la sous-section suivante contrecarrent cette attaque.

3.4.4 Vérifications *a posteriori*

L'uniformité de l'échantillonnage aléatoire du système dans les protocoles épidémiques est nécessaire à leur bon fonctionnement. De plus, elle doit être assurée dans un souci d'équité entre les différents utilisateurs du système. Pour ces raisons, LiFTinG effectue des vérifications statistiques sur l'historique des nœuds. D'autre part, l'historique d'un profiteur déjouant régulièrement les vérifications à l'aide d'un tiers contiendra bien plus souvent les identifiants des membres de sa coalition que les autres nœuds du système, s'écartant ainsi d'un échantillonnage uniforme. Intuitivement, les vérifications *a posteriori* permettent donc également de détecter les attaques aux vérifications instantanées.

Le multi-ensemble des nœuds avec qui un nœud p a interagi peut être extrait de son historique ou obtenu par recoupement de diverses manières. Tout d'abord on distingue le multi-ensemble des



(a) Vérification instantanée avec recoupement

(b) Attaque contre une vérification instantanée avec recoupement instantané à l'aide d'un tiers

FIGURE 3.4 : Vérification instantanée avec recoupement et attaque. Les profiteurs faisant partie de la même coalition sont notés avec un '$\star$'.

nœuds qui ont envoyé des blocs à p ($\mathcal{F}'_h$) de ceux à qui il en a envoyé ($\mathcal{F}_h$). La taille de ces deux multi-ensembles est $n_h f$. Le premier peut être extrait de l'historique de p ou en demandant aux nœuds à qui p a envoyé des blocs quels nœuds leur ont envoyé une demande de confirmation. De même, le second peut être obtenu directement ou en demandant aux nœuds qui ont envoyé des blocs à p les messages de liste renvoyés par p. La véracité de ces multi-ensembles peut être vérifiée par recoupement en interrogeant tout ou une partie des nœuds impliqués. Dans le cas d'un nœud honnête, ces méthodes donnent le même résultat. Pour un profiteur ayant attaqué les vérifications instantanées, ces multi-ensembles diffèrent suivant la méthode. Lorsque les multi-ensembles sont obtenus par recoupement, les membres de la coalition d'un tel profiteur apparaissent plus fréquemment que les autres. De même pour un profiteur ayant biaisé l'échantillonnage.

Une fois ces multi-ensembles obtenus, l'uniformité de l'échantillonnage est vérifiée à l'aide d'un test statistique d'adéquation. Un test d'adéquation consiste à tester l'hypothèse selon laquelle un échantillon d'une série statistique indépendante a été généré suivant une certaine loi de probabilité. Dans un cas discret comme celui considéré ici, la première étape consiste à estimer la distribution de probabilité en comptant le nombre d'apparitions de chaque valeur possible. Ici, il s'agit de compter le nombre d'occurrences $\{n_{h,i}\}_{i\in\{1,\dots,n\}}$ de chacun des nœuds dans les deux multi-ensembles $\mathcal{F}_h$ et $\mathcal{F}'_h$ extraits de l'historique comme expliqué dans le paragraphe précédent. On note $\tilde{d}_h$ la distribution de probabilité estimée. On a alors $\tilde{d}_{h,i} = n_{h,i}/(n_h f)$. Il existe plusieurs méthodes pour tester l'adéquation entre un échantillon et une loi de probabilité [Ste74].

Divergence de Kullback-Leibler [CT91] Cette divergence, issue de la théorie de l'information, permet d'estimer l'écart entre deux distributions de probabilité. Ici, il s'agit d'évaluer la similarité entre une loi de probabilité (la loi uniforme dans notre cas) et une distribution de probabilité empirique (ici $\tilde{d}_h$). Dans ce cas, le calcul de la divergence revient à calculer l'entropie de Shannon de la distribution empirique :

$$H(\tilde{d}_h) = -\sum_i \tilde{d}_{h,i} \log_2(\tilde{d}_{h,i}) \tag{3.1}$$

L'entropie est comprise entre 0 et $\log_2(n_h f)$ et est maximale lorsque tous les nœuds apparaissent le même nombre de fois. Dans le cas où n est plus grand que le nombre de nœuds apparaissant dans $\mathcal{F}_h$ et $\mathcal{F}'_h$, ce qui est le cas dans LiFTinG pour des raisons d'extensibilité, l'entropie est maximum quand chaque nœud apparaît au plus une fois dans le multi-ensemble. Le résultat doit alors être comparé à un seuil γ qui détermine la déviation maximale autorisée entre la loi et la distribution empirique, ce seuil devant prendre en compte les imperfections du générateur de nombres aléatoires utilisé par la méthode d'échantillonnage ainsi que de la méthode elle-même. Par exemple, avec le protocole d'échantillonnage CYCLON, l'uniformité est assurée en moyenne et l'indépendance des échantillons successifs est très faible mais non nulle. Cette considération doit également être prise en compte dans les méthodes de tests statistiques.

Test du χ^2 d'adéquation Ce test consiste à calculer une déviation moyenne entre l'échantillon et la loi de probabilité. En considérant que le processus ayant généré l'échantillon est aléatoire, cette valeur est également une variable aléatoire. Si la loi de probabilité de cette variable aléatoire est connue, on peut alors estimer la vraisemblance statistique de la réalisation observée. Dans le cas de ce test, pour un échantillonnage supposé uniforme, la déviation est une erreur quadratique moyenne calculée à l'aide de la formule suivante :

$$F_{\chi^2}(\tilde{d}_h) = \sum_{i=1}^{n_h f} (\tilde{d}_{h,i} - 1/n)^2 \tag{3.2}$$

et dans l'hypothèse où l'échantillon a bien été généré selon une loi uniforme, F_{χ^2} suit une loi du χ^2 à $n_h f - 1$ degrés de liberté. On utilise une table de dépassement pour mesurer le réalisme de l'hypothèse, c'est-à-dire les chances que l'utilisateur soit honnête.

Test de Kolmogorov-Smirnov Contrairement au précédent, ce test travaille sur la fonction de répartition empirique et mesure la déviation maximale entre celle-ci et la fonction de répartition de la loi de probabilité théorique (ici la loi uniforme). Dans le cas où l'échantillon est généré selon la loi de probabilité théorique, la loi de probabilité de cette divergence est connue (loi de Kolmogorov). On peut alors à nouveau quantifier la vraisemblance de l'hypothèse en utilisant une table de dépassement. Ce test est le plus utilisé, en particulier pour évaluer la qualité des générateurs de nombres aléatoires.

La méthode générale pour tester l'uniformité de l'échantillonnage à l'aide de l'historique des interactions d'un nœud est illustrée sur la figure 3.5. Dans LiFTinG, c'est la méthode entropique basée sur la divergence de Kullback-Leibler qui est utilisée et le seuil est déterminé par simulation comme expliqué dans la section suivante. En effet, les mécanismes entrant en ligne de compte sont trop complexes pour pouvoir être analysés de manière théorique ici (génération de nombres aléatoires, algorithme d'échantillonnage, etc.).

Si un nœud échoue au test statistique, il est exclu directement du système. En conclusion, un profiteur envoyant son historique sans le modifier a de grandes chances d'être exclu par les vérifications statistiques et un profiteur modifiant son historique pour passer le test statistique sera blâmé par le recoupement *a posteriori* et donc potentiellement exclu. Du fait de la sévérité des vérifications *a posteriori* et comme elles nécessitent de gros échanges d'informations (de l'ordre de $n_h f$), elles sont effectuées au moyen de connexions TCP. Cela permet en effet de communiquer de manière fiable, limitant ainsi les risques de perte de messages pouvant entraîner une exclusion injustifiée

du nœud. Le surcoût dû à l'établissement de la connexion est largement amorti par la quantité de données transférées. De plus, ces vérifications ne sont pas utilisées intensivement mais déclenchées de manière sporadique.

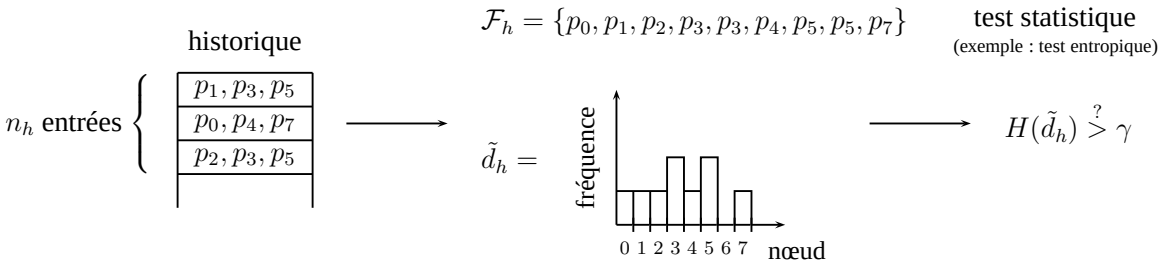


FIGURE 3.5 : Vérification statistique sur un historique de trois périodes et pour une arité sortante de deux ($n_h = 3$ et $f = 2$).

3.4.5 Surcoût de bande-passante

À cause des vérifications instantanées avec recoupement et des blâmes, LiFTinG entraîne un surcoût régulier de bande-passante. Pour déterminer une borne de ce surcoût, on détermine le nombre maximum de messages qu'un nœud envoie pour le bon fonctionnement de LiFTinG durant une période. Les vérifications *a posteriori* en revanche n'entraînent pas de surcoût régulier étant donné qu'elles sont déclenchées sporadiquement. Leur coût en nombre de messages est pour rappel $\mathcal{O}(n_h f)$ comme expliqué dans la sous-section précédente. Les résultats présentés dans les paragraphes suivants sont résumés dans le tableau 3.3.

Vérifications sans recoupement Ces vérifications ne nécessitent aucun échange de messages puisqu'il suffit au nœud ayant demandé des blocs de vérifier que ces derniers sont effectivement reçus. Cependant, ces vérifications peuvent conduire à l'émission de blâmes. Un nœud recevant f' ($f' \simeq f$ en moyenne) propositions par période et les messages de blâmes devant être envoyés aux M surveillants des nœuds visés, le surcoût en termes de messages de blâmes envoyés lors des vérifications sans recoupement est $\mathcal{O}(M \cdot f)$.

Vérifications avec recoupement Afin de vérifier que les blocs envoyés pendant la période précédente sont effectivement proposés, un nœud doit demander une confirmation à f nœuds, pour chacun des f nœuds à qui il a envoyé des blocs, et ce avec une probabilité p_{cc}. De la même manière, un nœud reçoit des demandes de confirmation de $p_{\text{cc}} \cdot f^2$ nœuds (en moyenne) auxquelles il doit répondre. Enfin un nœud doit également envoyer aux f nœuds (en moyenne) qui lui ont envoyé des blocs la liste des nœuds à qui il a proposé ces blocs. Contrairement aux demandes de confirmation, cette liste est envoyée à chaque période et non de manière probabiliste. Soit un total de $\mathcal{O}(p_{\text{cc}} \cdot f^2)$ messages. Aux messages nécessaires aux vérifications avec recoupement s'ajoutent les blâmes émis par ces dernières. Les blâmes envoyés par un nœud pendant une période concernent les f nœuds à qui il a envoyé des blocs durant la période précédente, et doivent être envoyés à leurs M surveillants respectifs. Soit un total de $\mathcal{O}(p_{\text{cc}} \cdot M \cdot f)$ messages de blâmes.

sans recoupement (messages)	0
sans recoupement (blâmes)	$\mathcal{O}(M \cdot f)$ (par le nœud exécutant la vérification)
avec recoupement (messages)	$\mathcal{O}(p_{\text{cc}} \cdot f^2)$ (par le nœud exécutant la vérification : demandes de confirmation) $\mathcal{O}(p_{\text{cc}} \cdot f)$ (par le nœud vérifié : liste) $\mathcal{O}(p_{\text{cc}} \cdot f^2)$ (par un nœud interrogé : réponse)
avec recoupement (blâmes)	$\mathcal{O}(p_{\text{cc}} \cdot M \cdot f)$ (pour le nœud exécutant la vérification)

TABLEAU 3.3 : Surcoût théorique de bande-passante des vérifications instantanées de LiFTinG.

En conclusion le surcoût régulier de LiFTinG, c'est-à-dire sans les vérifications sporadiques, est de $\mathcal{O}(f^2)$. En effet, p_{cc} est inférieur à 1 et M est une constante. Le nombre de messages envoyés par un nœud pour le protocole de diffusion épidémique lui-même est exactement $(2 + |\mathcal{S}|)f$: une proposition, une demande et $|\mathcal{S}|$ blocs pour chacun des f nœuds contactés. Il faut cependant tenir compte du fait que les messages tels que les propositions ou les demandes ne contiennent que quelques identifiants alors que les messages contenant des blocs représentent des données brutes de plusieurs milliers d'octets. Enfin on rappelle que l'arité sortante, f, est de l'ordre de $\ln n$ et donc que LiFTinG, au même titre que le protocole de diffusion, peut supporter un très grand nombre de nœuds.

3.5 Analyse et configuration de LiFTinG

Cette section présente une analyse théorique des performances de LiFTinG en fonction de ses différents paramètres. Cette analyse permet, d'une part, de paramétrer certaines composantes de LiFTinG de manière à ce que la détection soit optimale et, d'autre part, dans le cas où un compromis naturel apparaît entre les performance et le coût de LiFTinG, elle permet de choisir les valeurs des paramètres en connaissance de cause en fonction des exigences et des contraintes. Les résultats regroupés dans cette section se présentent sous forme d'expressions analytiques ou bornes pour l'harmonisation des scores, les seuils de détection et les probabilités de détection et de faux positifs. Ces résultats sont confirmés ou illustrés par des simulations numériques. Dans le cas où l'analyse dépend de paramètres inconnus ou de mécanismes complexes (générateur de nombres aléatoires par exemple), seuls les résultats des simulations sont présentés. Cette section est découpée en deux sous-sections. La première décrit le mécanisme de détection de LiFTinG basé sur les scores en tenant compte des pertes de messages et du modèle des profiteurs et la seconde traite du mécanisme de détection entropique en prenant en compte le degré de profit et le degré de favoritisme des profiteurs. Pour faciliter la lecture, le tableau 3.4 (page 43) récapitule les notations utilisées jusque-là ou introduites dans cette section.

3.5.1 Mécanisme de détection basé sur les scores

Les messages émis par le protocole épidémique à trois phases ainsi que ceux émis par les vérifications instantanées de LiFTinG sont envoyés via des connexions UDP et sont donc sujets à des pertes. Ces pertes peuvent conduire à l'émission de blâmes injustes, c'est-à-dire obtenus à tort par un nœud respectant le protocole. Par exemple, si certains blocs envoyés sont perdus par le canal de communication, l'expéditeur est blâmé, à tort, par les vérifications instantanées sans recoupement

effectuées par le destinataire des blocs. Les profiteurs quant à eux obtiennent des blâmes supplémentaires pour leurs déviations du protocole. Par conséquent, on s'attend à ce que la distribution des scores parmi les nœuds du système soit un mélange (*mixture* en anglais) de deux composantes correspondant respectivement aux nœuds honnêtes et aux profiteurs. Dans une telle situation, on utilise généralement des algorithmes de décision basés sur le maximum de vraisemblance. Ces algorithmes s'appuient sur les scores relatifs des nœuds pour décider lesquels sont des profiteurs, et ne sont donc pas sensibles aux blâmes injustes puisque ces derniers affectent le score de tous les nœuds de la même manière. Cependant plusieurs raisons empêchent d'utiliser de tels algorithmes de décision dans le contexte applicatif de LiFTinG.

Tout d'abord, il est important de noter que ces algorithmes nécessitent, a priori, la connaissance de la distribution globale des scores des nœuds dans le système. Cette information n'est pas accessible aux nœuds devant prendre la décision d'exclusion, à savoir les surveillants. D'autre part, comme évoqué dans le paragraphe définissant le modèle des profiteurs, les profiteurs ont intérêt à blâmer à tort des nœuds honnêtes si l'algorithme de décision se base sur des scores relatifs. En effet, dans ce cas, diminuer volontairement le score des nœuds honnêtes revient à augmenter le score des profiteurs. Enfin, le score des nœuds rejoignant le système n'est pas comparable à celui des nœuds déjà présents. Pour toutes ces raisons le mécanisme de détection de LiFTinG consiste à comparer le score harmonisé, c'est-à-dire translaté de la valeur moyenne des blâmes injustes dus aux pertes de messages et normalisé par le nombre de périodes passées dans le système, à un seuil fixe noté η. Chaque surveillant effectue cette opération sur le score des nœuds qu'il surveille et si le score ainsi harmonisé est inférieur à η, le surveillant initie une exclusion épidémique.

Afin de paramétrer le mécanisme d'harmonisation des scores nécessaire à l'algorithme de décision de LiFTinG, on fait l'hypothèse que les pertes de messages sont indépendantes et identiquement distribuées suivant une loi de Bernoulli de paramètre p_l et on calcule une expression analytique de l'espérance des blâmes injustes causés par les pertes de messages appliqués à un nœud honnête pendant une période. En réajustant périodiquement le score des nœuds de cette quantité, le score moyen des nœuds honnêtes est alors de zéro. Dans ce but, on analyse pour chacune des vérifications, les situations où des pertes de messages peuvent causer des blâmes injustes et on calcule leur valeur moyenne. On notera p_r la probabilité de réception ($p_r = 1 - p_l$). Cette probabilité peut être estimée une fois pour toute et codée en dur dans LiFTinG ou estimée dynamiquement et de manière répartie par les nœuds du système. Pour simplifier l'analyse, on suppose qu'un nœud reçoit au moins une proposition par période. Cette hypothèse est largement vérifiée en pratique. On suppose également qu'une proposition conduit à une demande d'un nombre fixe et commun à tous les nœuds $|\mathcal{R}|$ de blocs. Enfin on supposera également que les vérifications par recoupement sont toujours effectuées, c'est-à-dire $p_{\text{cc}} = 1$. Il est assez aisé d'intégrer ce paramètre dans les calculs mais cela nuit à la compréhension déjà délicate des calculs d'espérance nécessitant d'envisager tous les cas de perte possibles dans le protocole à trois phases auquel s'ajoutent les vérifications instantanées.

Vérification instantanée sans recoupement Considérons un nœud p_0 proposant des blocs à un nœud p_1. Si la proposition envoyée par p_0 est perdue, aucun blâme ne sera émis par p_1, ce dernier ne sachant pas qu'il était censé recevoir une proposition suivie de blocs de la part de p_0. Si la proposition est reçue (ce qui se produit avec une probabilité p_r), alors des blâmes risquent d'être émis.

Si la demande est perdue, aucun bloc n'est envoyé par p_0 et p_1 le blâme alors de f. Cette situation arrive avec une probabilité $p_r(1 - p_r)$ et donne le membre (a) de l'équation (3.3).

Si la demande est reçue, alors pour chaque bloc demandé par p_1 qu'il ne reçoit pas, p_1 blâme p_0 de $f/|\mathcal{R}|$. Cette situation arrive avec une probabilité p_r^2 et le nombre moyen de blocs envoyés par p_0 et non reçus par p_1 est de $(1-p_r)\,|R|$. Cela correspond au membre (b) dans l'équation (3.3). Un nœud envoyant des propositions à un total de f nœuds effectuant les mêmes vérifications que p_1, p_0 reçoit par les vérifications instantanées sans recoupement un blâme injuste total moyen de :

$$\tilde{b}_{\mathrm{dc}} = f \cdot \left[\overbrace{p_r(1-p_r)\cdot f}^{\text{(a)}} + \overbrace{p_r^2 \cdot (1-p_r)\,|\mathcal{R}| \cdot \frac{f}{|\mathcal{R}|}}^{\text{(b)}} \right] = p_r(1-p_r^2)\cdot f^2 \tag{3.3}$$

durant une période.

Vérification instantanée avec recoupement On considère un nœud p_0 ayant envoyé des blocs à un nœud p_1 lors de la dernière période, et on considère un nœud p_2 à qui p_1 envoie une proposition dans la période courante. On pourra se reporter pour rappel à la figure 3.3 (page 28) pour la description du protocole de vérification. Durant la dernière période, p_1 a reçu des propositions de f nœuds en moyenne, dont p_0. Parmi ceux-là une proportion p_r^2 lui a effectivement envoyé des blocs car il faut pour cela que leur proposition et la demande aient été reçues. Au total, $p_r^2 \cdot f$ nœuds vont donc effectuer une vérification instantanée avec recoupement sur p_1 durant cette période. D'autre part, chacune de ces procédures va contacter les nœuds à qui p_1 propose des blocs dans cette période, p_2 par exemple. Il y a f tels nœuds. On se concentre sur la vérification effectuée sur p_1 par p_0 en contactant p_2.

Si les blocs envoyés par p_0 n'ont pas tous été reçus par p_1 les propositions émises par p_1 sont incomplètes du point de vue de p_0 et il blâme donc p_1 de 1, f fois, indépendamment des autres échanges de messages (que p_1 ait effectivement contacté f nœuds ou pas, etc.). De même si la liste de nœuds envoyée par p_1 à p_0 est perdue. Cette situation arrive donc avec une probabilité $1-p_r^{|\mathcal{R}|+1}$. Cela correspond au membre (a) de l'équation (3.4).

Si p_1 a bien reçu tous les blocs envoyés par p_0 et que p_0 a bien reçu la liste envoyée par p_1 (cela arrive avec une probabilité $p_r^{|R|+1}$) alors des blâmes d'une valeur de 1 sont émis uniquement si la proposition envoyée par p_1 à p_2 est perdue ou si la demande de confirmation envoyée par p_0 à p_2 est perdue ou si la confirmation envoyée par p_2 à p_0 est perdue (cela arrive avec une probabilité $1-p_r^3$). Cette vérification concerne les f nœuds, tels que p_2, contactés par p_1. Ces blâmes correspondent au membre (b) de l'équation (3.4).

Le blâme injuste total moyen appliqué a un nœud honnête à cause des pertes de messages est donc de :

$$\tilde{b}_{\mathrm{cc}} = f \cdot p_r^2 \left[\overbrace{(1-p_r^{|R|+1})\cdot f}^{\text{(a)}} + \overbrace{f \cdot p_r^{|\mathcal{R}|+1}(1-p_r^3)}^{\text{(b)}} \right] = p_r^2(1-p_r^{|\mathcal{R}|+4})\cdot f^2 \tag{3.4}$$

par période.

Vérification *a posteriori* avec recoupement Cette procédure demande des confirmations aux nœuds apparaissant dans un historique pour en vérifier la véracité. Cette vérification peut également générer des blâmes injustes. En effet, comme pour les vérification instantanées, si un nœud n'a pas reçu la proposition du nœud inspecté, il ne confirmera pas l'entrée correspondante dans

l'historique, conduisant à un blâme de 1. Au total il y a $n_h f$ entrées dans l'historique et donc seulement $p_r n_h f$, en moyenne, seront confirmées, conduisant à un blâme injuste total moyen lors d'une vérification *a posteriori* de :

$$\tilde{b}_{\text{apcc}} = (1 - p_r) \cdot n_h f \ . \tag{3.5}$$

On notera cependant que les messages de recoupement étant envoyés via une connexion TCP, il n'y a pas de pertes de messages à prendre en compte de ce côté-là. Enfin, ces blâmes injustes ne doivent pas être corrigés périodiquement car ces vérifications ne sont déclenchées que sporadiquement lors d'une vérification *a posteriori*. Par conséquent, ils ne sont pas pris en compte par la procédure d'harmonisation décrite précédemment.

Résumé L'analyse précédente a permis, en considérant et en regroupant tous les cas possibles de pertes dans le protocole de diffusion à trois phases auquel s'ajoutent les vérifications, de calculer une expression analytique de l'espérance des blâmes injustes appliqué à un nœud honnête sur une période :

$$\tilde{b} = \tilde{b}_{\text{dc}} + \tilde{b}_{\text{cc}} = p_r(1 + p_r - p_r^2 - p_r^{|\mathcal{R}|+5}) \cdot f^2 \ . \tag{3.6}$$

En raisonnant d'une manière analogue on peut déduire une expression analytique de l'écart type $\sigma(b)$ de ces blâmes. La figure 3.6 représente la distribution des scores harmonisés dans un système de 10.000 nœuds honnêtes après une période. Ces résultats ont été obtenus par simulations en n'utilisant que des vérifications instantanées avec $p_{\text{cc}} = 1$. Le taux de pertes a été fixé à 7% (taux moyen constaté sur PlanetLab pour les paquets UDP) avec une arité sortante de 12 (soit légèrement plus que $\ln n$) et un nombre moyen de blocs demandés par proposition de $|\mathcal{R}| = 4$. Les scores ont été harmonisés en utilisant l'expression de l'équation (3.6) : $-\tilde{b} = 72,95$. On observe sur la figure que le score moyen (ligne verticale pointillée) est, comme prévu, proche de zéro (inférieur à 10^{-2} ici) ce qui signifie que les blâmes injustes ont été corrigés avec précision. Dans ces simulations, l'écart type des blâmes est $\sigma(b) = 25,6$.

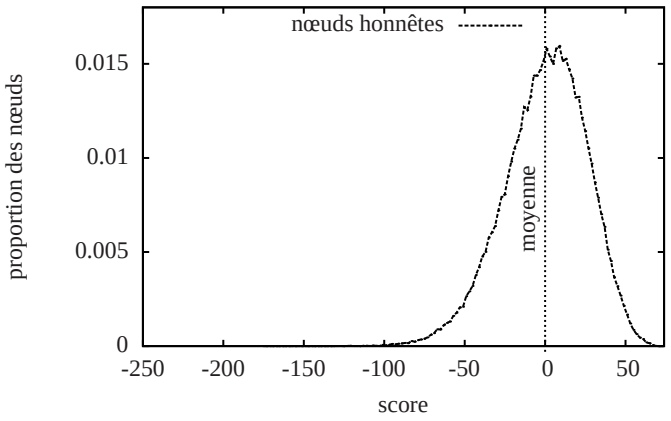


FIGURE 3.6 : Impact des pertes de messages sur le score des nœuds honnêtes.

On évalue maintenant les performances du mécanisme de détection de LiFTinG basé sur les scores : c'est-à-dire sa capacité à détecter les profiteurs et sa tendance à déclarer à tort des nœuds

honnêtes comme profiteurs. Ces performances sont caractérisées par la probabilité de détection (notée α) et la probabilité de faux positif notée (β). Dans la suite, on exhibe des bornes pour ces deux quantités.

Comme expliqué dans l'en-tête de cette section, un nœud est détecté et exclu lorsque son score harmonisé devient inférieur à un seuil η. Formellement, le score harmonisé d'un nœud honnête à la période t s'écrit comme une fonction du nombre r de périodes qu'il a passées dans le système, les blâmes $(b_i)_{i\in\{t-r,\ldots,t\}}$ qu'il a reçus à cause des pertes de messages au cours de ces différentes périodes et de la correction appliquée pour les blâmes injustes :

$$s = -\frac{1}{r}\sum_{i=0}^{r}(b_{t-i} - \tilde{b}), \tag{3.7}$$

L'analyse menée sur les blâmes injustes permet de calculer leur espérance et leur écart type. En supposant que ces blâmes soient indépendants et identiquement distribués, le score d'un nœud honnête est une variable aléatoire d'espérance nulle $\mathbb{E}[s] = 0$ et d'écart type $\sigma(s) = \sigma(b)/\sqrt{r}$ (car l'espérance est linéaire et la variance quadratique). En utilisant l'inégalité de Bienaymé-Tchebychev, on obtient une borne supérieure sur la probabilité de faux positif, c'est-à-dire la probabilité que le score harmonisé d'un nœud honnête soit plus petit que η ($\eta < 0$) :

$$\beta = P\left(s < \eta\right) \leq P\left(|s| > -\eta\right) \leq \frac{\sigma(b)^2}{r \cdot \eta^2} \tag{3.8}$$

La probabilité de détecter un profiteur dépend de l'amplitude de sa déviation au protocole, appelé degré de profit. On définit cette notion formellement à l'aide d'un triplet $\Delta = (\delta_1, \delta_2, \delta_3)$, $0 \leq \delta_1, \delta_2, \delta_3 \leq 1$ tel qu'un profiteur de degré de profit Δ ne contacte que $(1-\delta_1)\cdot f$ nœuds par période, ne propose qu'une proportion $(1-\delta_2)$ des blocs reçus (plus précisément ne propose que les blocs envoyés par une proportion $(1-\delta_2)$ des nœuds qui lui ont proposé des blocs) et n'envoie réellement que $(1-\delta_3)\cdot|\mathcal{R}|$ des blocs demandés. Le gain d'un tel profiteur est une fonction de son degré de profit : $1-(1-\delta_1)(1-\delta_2)(1-\delta_3)$.

En suivant la même méthodologie que pour le calcul de l'espérance des blâmes injustes dûs aux pertes de messages, on calcule l'espérance des blâmes $\tilde{b}(\Delta)$ émis contre un profiteur de degré Δ. Ces derniers tiennent compte des pertes de messages auxquelles les messages envoyés par les profiteurs sont également sujets. Bien que légèrement plus complexe, la technique reste la même. Pour les vérifications sans recoupement, la probabilité qu'un bloc demandé ne soit pas envoyé devient $(1-p_r^2(1-\delta_3))$ et le nombre de nœuds effectuant cette vérification devient $(1-\delta_1)\cdot f$. Pour les vérifications avec recoupement, la probabilité qu'une proposition soit complète du point de vue d'un vérificateur devient $(1-\delta_2)p^{|\mathcal{R}|}$. Enfin, la probabilité qu'un nœud interrogé ne réponde pas ou réponde négativement à la confirmation devient $(1-p_r^3(1-\delta_1))$. On notera que le résultat obtenu précédemment dans l'équation (3.6) correspond à celui de l'équation (3.9) pour un degré de profit nul.

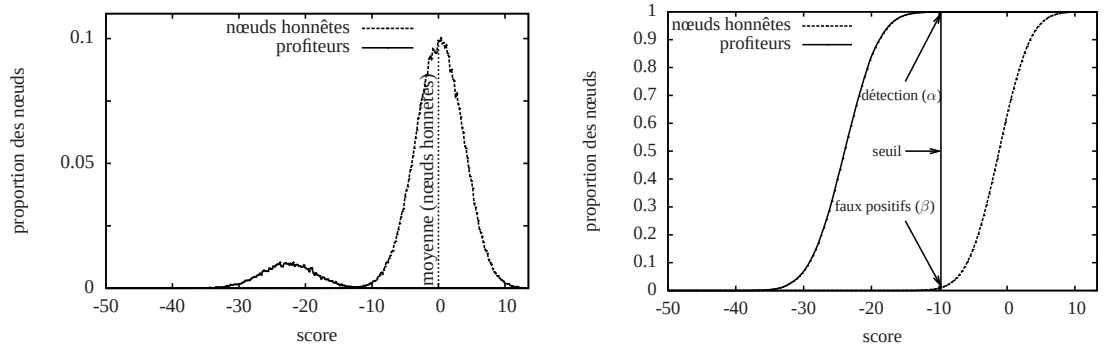
$$\begin{aligned}\tilde{b}'(\Delta) \;=\; & (1-\delta_1)\cdot p_r\left(1-p_r^2(1-\delta_3)\right)\cdot f^2 + \\ & p_r^2\cdot\left[(1-\delta_2)p_r^{|\mathcal{R}|+1}(1-p_r^3(1-\delta_1)) + (1-(1-\delta_2)p_r^{|\mathcal{R}|+1})\right]\cdot f^2\end{aligned} \tag{3.9}$$

À nouveau, l'écart type $\sigma(b(\Delta))$ peut être calculé (ou estimé par simulation) et on peut alors trouver une borne inférieure à la probabilité de détection en fonction du degré de profit des profiteurs :

$$\alpha \geq 1 - \frac{\sigma(b'(\Delta))^2}{r \cdot (\tilde{b'}(\Delta) - \eta)^2} \tag{3.10}$$

Dans l'analyse d'homogénéité des blâmes on notait que les trois déviations possibles menaient au même blâme pour un même gain. Par conséquent, un profiteur pouvait les implémenter indistinctement. Cependant, un profiteur peut potentiellement obtenir un blâme plus petit pour un certain gain en combinant astucieusement les déviations. Cela revient à minimiser la probabilité de détection $\alpha(\Delta)$ (ou minimiser la borne inférieure de l'équation (3.10)) sous la contrainte $1-(1-\delta_1)(1-\delta_2)(1-\delta_3) = g$ où g est le gain souhaité. Dans un souci de simplicité, on considère que les profiteurs implémentent les trois déviations dans les mêmes proportions.

On notera que sous l'hypothèse d'indépendance des blâmes émis, les performances de LiFTinG sont asymptotiquement optimales. Étant donné que le seuil de détection est fixe, la probabilité de détection α tend vers 1 et la probabilité de faux positif tend vers 0 quand le nombre r de périodes passées dans le système tend vers l'infini.



(a) densité de probabilité des nœuds honnêtes et des profiteurs

(b) fonction de répartition et illustration des critères de performance

FIGURE 3.7 : Distribution des scores harmonisés en présence de profiteurs de degré $\Delta = (0.1, 0.1, 0.1)$.

La figure 3.7 représente la distribution des scores harmonisés des nœuds dans un système de 10.000 nœuds dont 1.000 profiteurs de degré $\Delta = (0.1, 0.1, 0.1)$ après $r = 50$ périodes. La distribution des scores est représentée, d'une part, parmi les nœuds honnêtes et parmi les profiteurs d'autre part (la seconde distribution a été renormalisée pour refléter la proportion de profiteurs dans le système). Comme prévu, la distribution de scores est constituée de deux modes avec un faible recouvrement (voir figure 3.7a) : le mode le plus à gauche correspondant aux scores les plus faibles est celui des profiteurs et le mode de droite est celui des nœuds honnêtes, à nouveau de moyenne proche de zéro. Cette faible déviation de la moyenne (légèrement inférieure à zéro) est due à la présence des profiteurs dont les déviations faussent légèrement les calculs des blâmes injustes fait dans les paragraphes précédents. Le fossé séparant les deux modes est clairement identifiable ce qui est synonyme d'un fort potentiel pour de bonnes performances (c'est-à-dire qu'il existe une possibilité de détecter une grande proportion de profiteurs sans pour autant détecter à tort des nœuds honnêtes), même si ce n'est pas sur l'analyse de cette courbe elle-même qu'est basé l'algorithme de détection. La figure 3.7b représente la fonction de répartition des scores harmonisés et illustre les

notions de détection et de faux positifs mentionnés plus haut pour un seuil donné (ici $\eta = -9,75$). Cette valeur de seuil donne un taux de faux positifs inférieur à 1%. On peut donc fixer le seuil à cette valeur (indépendamment des profiteurs). Les performances en termes de détection dépendent alors du degré de profit des profiteurs. On réalise plusieurs simulations pour des degrés de profits différents. Les résultats d'un déploiement réel sont rapportés dans la prochaine section. On suppose que les profiteurs implémentent les trois déviations dans les même proportions, c'est-à-dire $\delta_1 = \delta_2 = \delta_3 = \delta$ et on observe la proportion de profiteurs détectés par LiFTinG après 50 périodes. La figure 3.8 représente α et le gain des profiteurs en fonction de δ. Par exemple, on peut voir qu'un profiteur de degré 5%, correspondant à une économie de bande-passante montante de 15%, est détecté dans 65% des cas. Au delà de 10% il est détecté dans plus de 99% des cas. L'hypothèse faite dans FlightPath [LCM$^+$08] sur ce qu'est un gain significatif est qu'un utilisateur est prêt à faire l'effort de télécharger et installer un client alternatif (implémentant les déviations permettant de diminuer sa contribution) si ce dernier lui apporte un gain d'au moins 10%. Dans le protocole épidémique considéré ici, un gain de 10% correspond à un degré de $3,5\%$, degré pour lequel la probabilité de détection de LiFTinG est de 50% (voir figure 3.8).

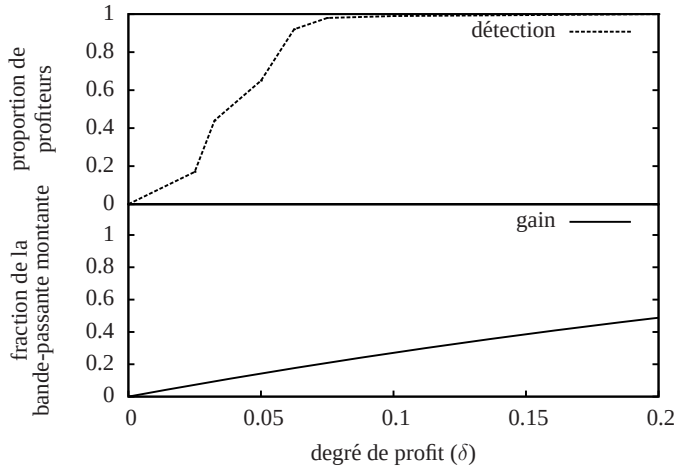


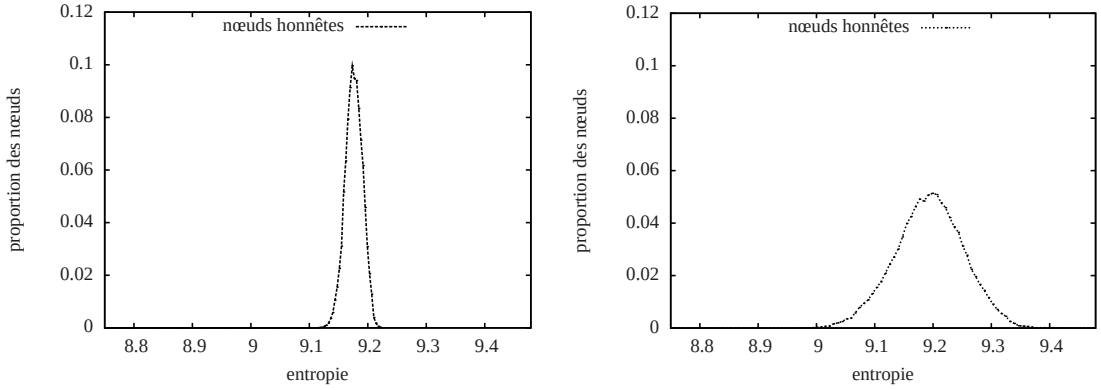
FIGURE 3.8 : Proportion des profiteurs détectés par LiFTinG.

3.5.2 Mécanisme de détection entropique

Dans un souci d'équité et pour contrer les attaques contre les vérifications instantanées par recoupement, LiFTinG comporte un mécanisme d'audit de l'historique des nœuds consistant en une vérification entropique pouvant mener directement à l'exclusion du nœud audité. Une vérification entropique consiste à calculer l'entropie de Shannon des ensembles $\mathcal{F}_h$ et $\mathcal{F}'_h$ de nœuds avec qui le nœud audité a interagi durant les dernières périodes et à la comparer à un seuil fixe. La distribution de l'entropie de l'historique des nœuds du système dépend principalement de la méthode d'échantillonnage et du générateur de nombres aléatoires utilisé par celle-ci. On notera que le générateur de nombres aléatoires dépend du langage utilisé et d'autres paramètres du système d'exploitation ou de la machine virtuelle utilisée pour JAVA par exemple. Pour ces raisons, on estime cette distribution par simulation et on détermine le seuil γ utilisé dans LiFTinG en se basant sur ces résultats. Dans la

suite, on considère un échantillonnage utilisant la liste complète des nœuds du système d'une part car cette méthode, bien que peu pratique, est optimale et parce qu'il a été montré que CYCLON, par exemple, avait des performances très proches de celle-ci [JVG+07].

La figure 3.9 représente la distribution d'entropie pour un historique de $n_h f = 600$ nœuds ($n_h = 50$ et $f = 12$) dans un système de 10.000 nœuds où la méthode d'échantillonnage utilise la liste complète des nœuds et est implémentée en JAVA. Les sous-figures 3.9a et 3.9b concernent respectivement les multi-ensembles $\mathcal{F}_h$ et $\mathcal{F}'_h$. Pour le premier, l'entropie observée varie entre $9,11$ et $9,21$ pour une valeur théorique maximale de $\log_2(n_h f) = 9,23$. La seconde est plus étalée et varie entre entre $8,98$ et $9,34$. Cette variation est due au fait que l'arité entrante est une variable aléatoire de moyenne f alors que l'arité sortante est fixe (et exactement égale à f). Par conséquent, l'entropie de $\mathcal{F}'_h$ peut dépasser la borne théorique qui s'applique à $\mathcal{F}_h$. En se basant sur les résultats de simulations, on voit qu'en fixant la valeur du seuil γ à $8,95$, le taux de faux positif est inférieur à 1%.



(a) entropie de sortie

(b) entropie d'entrée

FIGURE 3.9 : Distribution de l'entropie pour un échantillonnage implémenté en JAVA utilisant la liste complète des nœuds.

Une fois le seuil fixé on peut déterminer le gain maximum qu'un profiteur peut tirer du protocole sans être détecté par les vérifications entropiques. On considère une coalition de taille $m' + 1$, soit m' complices pour un profiteur donné, par exemple un groupe d'amis utilisant le même protocole pour regarder un flux diffusé sur Internet et souhaitant s'entraider. On formalise le biais introduit par les profiteurs dans la méthode d'échantillonnage, que ce soit pour n'envoyer du contenu qu'aux membres de leur coalition ou pour déjouer les vérifications instantanées avec recoupement, à l'aide d'un degré de favoritisme noté p_m, $0 \leq p_m \leq m'/n$. Lorsqu'il choisit les nœuds à qui envoyer des propositions, un profiteur réalise tout d'abord un tirage aléatoire de Bernoulli de paramètre p_m. Avec une probabilité p_m, il décide de choisir parmi les profiteurs de sa coalition (le choix est ensuite fait en suivant une loi de probabilité p_X parmi eux), sinon il choisit selon une loi p_Y parmi les nœuds honnêtes du système. Un facteur de favoritisme de 1 consiste à ne choisir que des profiteurs et un facteur de m'/n revient à réaliser un échantillonnage équitable du système. Dans la suite on cherche à déterminer les lois p_X et p_Y donnant une entropie maximale, un profiteur cherchant à minimiser la probabilité d'être détecté et exclu. On notera qu'un premier pré-requis pour que les vérifications entropiques soient efficaces est que le nombre d'entrées dans l'historique (c'est-à-dire

$n_h f$) soit supérieur au nombre de profiteurs dans la coalition. Sinon, en ne proposant successivement qu'à chacun de ses amis (en utilisant l'algorithme du tourniquet), un profiteur obtiendrait un historique d'entropie maximale et ne pourrait donc pas être détecté. On suppose donc $n_h f \gg m'$. De plus, pour des raisons d'extensibilité, on impose $n_h f \ll n$. Pour déterminer les lois p_X et p_Y on exprime l'entropie d'une entrée de l'historique en fonction d'elles en utilisant le fait que X et Y sont indépendantes :

$$
\begin{aligned}
H(\mathcal{F}_h) &= -\sum_{n\in X\cup Y} p(n)\log_2(p(n)) \\
&= -\sum_{n\in X} p(n)\log_2(p(n)) - \sum_{n\in Y} p(n)\log_2(p(n)) \\
&= -\sum_{n\in X} p_m p_X(n)\log_2(p_m p_X(n)) - \sum_{n\in Y}(1-p_m)p_Y(n)\log_2((1-p_m)p_Y(n)) \\
&= -p_m\sum_{n\in X}\left(p_X(n)\log_2(p_X(n)) + p_X(n)\log_2(p_m)\right) \\
&\quad -(1-p_m)\sum_{n\in Y}\left(p_Y(n)\log_2(p_Y(n)) + p_Y(n)\log_2(1-p_m)\right) \\
&= -p_m\left(\log_2(p_m)\underbrace{\sum_{n\in X} p_X(n)}_{=1} + \underbrace{\sum_{n\in X}\log_2(p_X(n))\cdot p_X(n)}_{=-H(X)}\right) \\
&\quad -(1-p_m)\left(\log_2(1-p_m)\underbrace{\sum_{n\in Y} p_Y(n)}_{=1} + \underbrace{\sum_{n\in Y}\log_2(p_Y(n))\cdot p_Y(n)}_{-H(Y)}\right) \\
H(\mathcal{F}_h) &= -p_m\log_2 p_m - (1-p_m)\log_2(1-p_m) + p_m H(X) + (1-p_m)H(Y)\ . \qquad (3.11)
\end{aligned}
$$

Pour une valeur de p_m fixée on observe que l'entropie de l'historique est maximale quand $H(X)$ et $H(Y)$ le sont, c'est-à-dire quand le choix parmi les nœuds honnêtes et le choix parmi les profiteurs sont uniformes. Dans ce cas, les entropies de X et de Y s'expriment simplement. Un profiteur peut faire en sorte qu'un nœud honnête apparaisse au plus une fois dans son historique car le nombre d'entrées dans l'historique est largement inférieur au nombre de nœuds honnêtes. L'entropie des entrées de l'historique contenant un nœud honnête (soit $(1-p_m)n_h f$ entrées) peut donc être rendue maximale : $H(Y) = \log_2((1-p_m)n_h f)$. En revanche les profiteurs apparaissent plus d'une fois dans l'historique car il y a bien moins de profiteurs que d'entrées dans l'historique. La fréquence d'apparition est la même pour tous les profiteurs, à savoir $1/m'$, et l'entropie de X est donc : $H(X) = -\log_2(m')$. On obtient donc une relation entre le seuil γ, la taille de la coalition $m'+1$ et le degré de favoritisme maximal $p_m^\star$ qu'un profiteur puisse utiliser sans être détecté par les vérifications entropiques :

$$\begin{aligned}\gamma &= -p^\star_m \log_2 p^\star_m - (1-p^\star_m)\log_2(1-p^\star_m) \\ &\quad -p^\star_m \log_2\left(\frac{1}{m'}\right) - (1-p^\star_m)\log_2\left(\frac{1}{(1-p^\star_m)n_h f}\right) \\ \gamma &= -p^\star_m \log_2\left(\frac{p^\star_m}{m'}\right) - (1-p^\star_m)\log_2\left(\frac{1}{n_h f}\right) \qquad (3.12)\end{aligned}$$

En inversant l'équation (3.12) numériquement pour $\gamma = 8,95$ comme choisi dans le paragraphe précédent, on obtient que le degré de favoritisme maximal pour un profiteur appartenant à une coalition de 25 nœuds est de 15%. Cela signifie qu'un utilisateur peut dédier 15% de sa bande-passante montante à envoyer des blocs à ses amis ou économiser 15% de sa bande-passante montante en demandant à ses amis de le couvrir. Cette proportion est tout à fait raisonnable. En effet eMule, un client très populaire pour le système de partage de fichiers eDonkey, autorise par exemple un utilisateur à dédier jusqu'à 20% de sa bande-passante montante pour ses amis.

Notation	**Description**
n, m	nombre de nœuds (respectivement de profiteurs)
$\|\mathcal{R}\|$, $\|\mathcal{S}\|$	nombre de blocs respectivement demandés et envoyés
f	arité sortante
n_h	taille de l'historique en nombre de périodes
$\mathcal{F}_h, \mathcal{F}'_h$	multi-ensemble des nœuds à qui des blocs ont été envoyés (respectivement, par qui des blocs ont été proposés)
p_{cc}	probabilité de déclencher une vérification par recoupement
p_l , p_r	probabilité de perte (respectivement de réception) ($p_r = 1 - p_l$)
$\tilde{b}$	espérance des blâmes injustes (à cause des pertes de messages)
$\sigma(b)$	écart-type des blâmes injustes
r	nombre de périodes passées dans le système
s	score harmonisé
$\Delta = (\delta_1, \delta_2, \delta_3)$	degré de profit
$\tilde{b}(\Delta)$	espérance des blâmes obtenus par un profiteur
$\sigma(b(\Delta))$	écart-type des blâmes obtenus par un profiteur
η	seuil de détection pour la détection basée sur les scores
α	probabilité de détection pour la détection basée sur les scores
β	probabilité de faux positif pour la détection basée sur les scores
γ	seuil de détection pour la détection entropique

TABLEAU 3.4 : Résumé des principales notations utilisées.

3.6 Évaluation et résultats expérimentaux

Cette section décrit le contexte du déploiement et de l'évaluation de LiFTinG et en rapporte les résultats. LiFTinG a été intégré dans un système complet et opérationnel de diffusion épidémique de flux vidéo développé par Maxime Monod [FGK+09b], l'un des membres du projet LiFTinG.

Ce système a été implémenté en Java et fonctionne comme une tâche de fond interfacé avec le lecteur vidéo VLC. À la source, VLC génère et transmet au programme de diffusion épidémique des blocs vidéo de type MPEG pouvant être contenus dans un paquet UDP. Ces paquets sont alors injectés dans le système. Chaque utilisateur du système exécute un programme similaire implémentant le protocole de dissémination de contenus servant de base à ce travail et incluant LiFTinG. La séquence réordonnée des blocs reçus est transmise à l'instance de VLC lancée sur l'ordinateur de l'utilisateur pour la lecture du flux vidéo. D'un point de vue technique, le protocole est implémenté de manière événementielle à l'aide de la librairie d'ordonnancement GenericRuntime [FM10] développée en parallèle du système de diffusion. L'aspect périodique du protocole épidémique est donc implémenté en programmant régulièrement l'envoi d'un paquet de propositions. La méthode d'échantillonnage utilisée est celle de CYCLON, implémentée comme un module de la librairie GossipLib [FM10] également développée en parallèle de ce système. Le déploiement du système de diffusion a été effectué sur la plate-forme de recherche PlanetLab. Celle-ci comporte plusieurs centaines d'ordinateurs répartis dans des instituts de recherche et universités du monde entier. Ces ordinateurs disposent d'une grande capacité en termes de calculs et de bande-passante. Cependant, il s'agit d'une ressource partagée par les chercheurs du monde entier et il est donc fréquent que les ordinateurs utilisés soient surchargés lors des expérimentations. Pour que les résultats expérimentaux soient fidèles à un déploiement réel pour le grand public principalement connecté à Internet par des connexions ADSL, une couche logicielle a été ajoutée pour limiter la bande-passante montante des nœuds. La limite a été fixée légèrement au-dessus du débit du flux vidéo diffusé.

3.6.1 Contexte expérimental

LiFTinG a été déployé sur 300 ordinateurs de la plate-forme PlanetLab où un flux d'un débit de 674 kbps est diffusé par une unique source. La période du protocole épidémique a été fixée à 500 millisecondes et l'arité sortante à 7. Parmi les 300 nœuds, 30 sont des profiteurs implémentant dans les même proportions les trois déviations présentées dans les sections précédentes. Ces derniers ne contactent que 6 nœuds par période, ne proposent que 90% des blocs reçus et n'envoient que 90% des blocs demandés, c'est-à-dire $\Delta = (1/7, 9/10, 9/10)$. Le gain émanant de cette combinaison de déviations en termes de bande-passante montante économisée est légèrement supérieur à 30%. L'architecture de blâme utilise $M = 25$ surveillants par nœud.

3.6.2 Surcoût pratique de bande-passante

Le tableau 3.5 rassemble les surcoûts de bande-passante générés par LiFTinG pour différentes valeurs de la fréquence de déclenchement des vérifications instantanées avec recoupement p_{cc}. On remarquera que le surcoût n'est pas nul, mais devient négligeable, quand cette fréquence est fixée à zéro car le message de liste est toujours envoyé et les blâmes également. Le surcoût reste raisonnable même lorsque les vérifications instantanées avec recoupement sont déclenchées durant chaque période.

3.6.3 Résultats expérimentaux

Pour évaluer LiFTinG, les expérimentations ont été effectuées pour deux valeurs de la fréquence de déclenchement des vérifications instantanées avec recoupement : $p_{\text{cc}} = 0,5$ et $p_{\text{cc}} = 1$. Tous les nœuds ont été introduits dans le système au même instant (temps zéro). La figure 3.10 représente les

Vérifications	**Instantanées**			**A posteriori**
Paramètres	$p_{cc} = 0$	$p_{cc} = 0,5$	$p_{cc} = 1$	$n_h = 50$
Flux d'un débit de 674 kbps	$1,07\%$	$4,53\%$	$8,01\%$	$3,60\%$
Flux d'un débit de 1082 kbps	$0,69\%$	$3,51\%$	$5,04\%$	$2,89\%$
Flux d'un débit de 2036 kbps	$0,38\%$	$2,80\%$	$2,76\%$	$1,74\%$

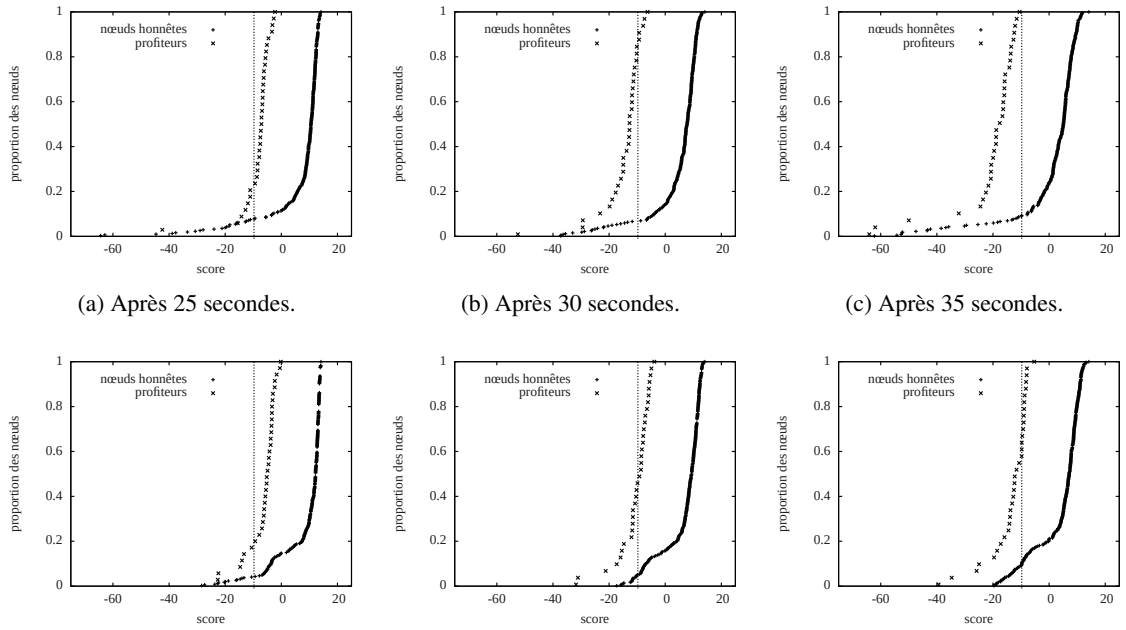
TABLEAU 3.5 : Surcoût pratique de bande-passante généré par LiFTinG.

fonctions de répartition des scores pour les profiteurs ($\times$) et les nœuds honnêtes ($+$), pour différentes valeurs de la fréquence des vérifications instantanées avec recoupement (première ligne pour $p_{\text{cc}} = 1$ et seconde ligne pour $p_{\text{cc}} = 0,5$) à trois instants donnés (après, de gauche à droite, 25, 30 et 35 secondes). Le seuil de détection η est représenté par une ligne pointillée verticale et fixé à $-9,75$ comme spécifié dans l'analyse. La proportion de profiteurs détectés par LiFTinG se lit à l'ordonnée du point d'intersection entre la fonction de répartition correspondante et du seuil. De même, le taux de faux positifs se lit à l'intersection de la fonction de répartition des scores des nœuds honnêtes et le seuil. Les scores ont été harmonisés comme expliqué dans la section précédente, en estimant le taux de pertes des paquets UDP.

Les fonctions de répartition des scores des profiteurs et des nœuds honnêtes sont clairement séparées. On peut voir sur la figure 3.10b qu'après seulement 30 secondes, 86% des profiteurs sont exclus et 12% des nœuds honnêtes le sont également, à tort. Ces nœuds honnêtes ont en réalité de faibles capacités en termes de bande-passante et de calcul entraînant une connectivité intermittente, des pertes de messages, etc. En d'autres termes, leur comportement est très proche de celui des profiteurs, à ceci prêt qu'il n'est pas volontaire, et ils sont par conséquent détectés comme tels par LiFTinG. Leur exclusion est acceptable dans la mesure où des nœuds n'ayant pas la capacité de participer à la diffusion ne devraient pas être autorisés à rejoindre le système en premier lieu, à moins d'utiliser un mécanisme perfectionné de répartition de charge tel que HEAP [FGK$^+$09a], mais cela rend l'utilisation de LiFTinG plus délicate car l'arité sortante varie alors selon les capacités du nœud. Comme on pouvait s'y attendre, le mécanisme de détection est plus efficace pour une fréquence de vérifications instantanées avec recoupement plus importante. Cependant pour $p_{\text{cc}} = 0,5$ la détection n'est pas deux fois plus lente, une partie des vérifications instantanées étant toujours effectuée, à savoir celles sans recoupement. Ces dernières contribuent activement à la détection car δ_3 est strictement positif. On observe ainsi qu'après 35 secondes avec une fréquence de $0,5$ (figure 3.10f) les résultats sont comparables à ceux obtenus après 30 secondes pour une fréquence de 1 (figure 3.10b).

Comme prévu dans l'analyse, le fossé séparant les deux fonctions de répartition se creuse au cours du temps. Cependant, la variance ne diminue pas (la pente de la fonction de répartition). Cela vient du fait que l'hypothèse d'indépendance des blâmes émis durant des périodes successives n'est pas toujours respectée et que le taux de pertes de messages n'est le même pour tous les nœuds.

Par conséquent, le caractère i.i.d. des blâmes est remis en cause en pratique, les problèmes de connexion ou de surcharge de l'ordinateur d'un utilisateur durant en général plusieurs périodes successives, voire l'intégralité de l'expérience.



(a) Après 25 secondes. (b) Après 30 secondes. (c) Après 35 secondes.

(d) Après 25 secondes. (e) Après 30 secondes. (f) Après 35 secondes.

FIGURE 3.10 : Fonction de répartition des scores pour $p_{\text{cc}} = 1$ (première ligne) et $p_{\text{cc}} = 0,5$ (deuxième ligne).

3.7 Résumé et perspectives

Ce chapitre a décrit la conception de LiFTinG, un protocole léger de détection des profiteurs pour les systèmes de diffusion épidémique de contenus volumineux. LiFTinG est complètement opérationnel comme le prouve son intégration dans un système existant et son déploiement dans un environnement réel. De plus, il peut être configuré de manière rigoureuse grâce à l'analyse théorique présentée dans ce chapitre. Le concept de vérifications réparties exploitant l'aspect aléatoire des protocoles épidémiques semble être une voie prometteuse pour la détection fiable des tricheurs dans les systèmes à grande échelle. D'autre part, il est envisageable que les méthodes employées par LiFTinG soient transposées à la classe orthogonale des systèmes à échanges symétriques, élargissant ainsi la portée de ce travail à la quasi-totalité des systèmes réels d'échange de contenus déployés actuellement. Par exemple, on peut imaginer un système à échanges symétriques utilisant un mécanisme donnant-donnant pour se prémunir des profiteurs, dont la composante asymétrique, c'est-à-dire l'approvisionnement opportuniste, serait sécurisée par LiFTinG.

CHAPITRE 4

RÉSEAUX SOCIAUX

Cette partie est consacrée à la sécurité des calculs répartis en présence de tricheurs lorsque les entités participant au calcul sont membres d'un réseau social. Certaines parties de ce travail ont été réalisées en collaboration avec Rachid Guerraoui (EPFL), Maxime Monod (EPFL), Ýmir Vigfússon (IBM Research Haifa) et Andrei Giurgiu (EPFL). Les résultat principaux présentés dans cette partie sont l'introduction d'un modèle de fautes pour les utilisateurs d'un réseau social tenant compte du fait que les tricheurs (1) ont un but précis et (2) se soucient de leur réputation, la présentation de DPol – un protocole de sondage binaire fournissant dans ce modèle des garanties de confidentialité et de précision (section 4.2), la formalisation du problème de calculs sécurisés et passant à l'échelle dans un réseau social (problème S^3 pour *Scalable and Secure computation in Social networks*, section 4.3) et enfin la conception du protocole Agg-S^3 résolvant le problème S^3 sous certaines hypothèses de régularité de la fonction à calculer (section 4.4).

Par opposition aux réseaux pair-à-pair où l'anonymat relatif des entités incitait à la tricherie et rendait nécessaire l'exclusion pure et simple des tricheurs, la logique de ce chapitre est d'exploiter le lien entre une entité participant au calcul et un utilisateur réel clairement identifié par son profil dans le réseau social. L'idée de base reste cependant la même que dans le chapitre précédent : procéder par dissuasion et détecter les déviations à l'aide de vérifications réparties. L'approche utilisée dans ce travail est d'utiliser des mécanismes de vérifications réparties conduisant à l'annotation du profil des utilisateurs dans le réseau social. En supposant que les utilisateurs se soucient de leur réputation, ils ne commettent pas de déviations au protocole de calculs répartis pouvant entrainer l'ajout de remarques négatives à leur propos qui compromettraient leur réputation. Cela réduit donc la classe de déviations que ces derniers implémentent permettant ainsi de fournir des garanties sur la précision du calcul. Les garanties de confidentialité sont assurées à l'aide de techniques de partage de secrets. Bien que les résultats présentés dans ce chapitre soient principalement théoriques, une des motivations principales de ce travail est de concevoir des systèmes pratiques. Cette volonté se manifeste à travers des propriétés d'extensibilité (c'est-à-dire capacité à gérer un grand nombre de participants) et de sécurité (c'est-à-dire précision et confidentialité) nécessaires respectivement à un déploiement efficace pour plusieurs centaines de millions d'utilisateurs et une adoption par le grand

public. L'aspect pratique des protocoles présenté est démontré par un déploiement PlanetLab.

La présentation de ce travail à la communauté scientifique s'est fait en plusieurs étapes. Une description du modèle d'utilisateurs de réseaux sociaux et les perspectives d'application au calcul réparti ont été présentées sous forme d'une brève annonce à la conférence DISC 2009 (23rd International Symposium on Distributed Computing) [GHKM09c]. Le protocole DPol apparaît dans les actes de la conférence OPODIS 2009 (12th International Conference on Principles of Distributed Systems) [GHKM09a] où il a obtenu le prix du meilleur article étudiant et Agg-S^3 apparaît dans les actes de la conférence SSS 2010 (12th International Symposium on Stabilization, Safety and Security of Distributed Systems) [GGHK10].

4.1 Introduction

Ces dernières années ont vu naître, puis rapidement exploser, les réseaux sociaux en ligne. Le nombre d'utilisateurs des réseaux sociaux, après une phase de croissance exponentielle, continue de grandir régulièrement aujourd'hui. L'un des plus populaires d'entre eux, Facebook [Fac], se targue aujourd'hui de rassembler plus de 500 millions d'utilisateurs. Les réseaux sociaux en ligne constituent de gigantesques plate-formes collaboratives sur lesquelles les utilisateurs, souvent directement identifiés par leur nom et prénom via un profil, déclarent leurs relations d'amitié, familiales ou professionnelles. Plus formellement, un réseau social en ligne est un graphe (dont les sommets sont les utilisateurs et les arêtes les relations sociales) et les actions collaboratives des utilisateurs sont des calculs. Par exemple, un sondage est le calcul, pour chacune des options proposées, du nombre d'utilisateurs choisissant cette option. Si on considère un groupe d'amis organisant une soirée et initiant un sondage binaire pour décider si les enfants sont conviés (ou plutôt tolérés), il apparaît clairement que les utilisateurs (1) ne veulent pas révéler leur préférence et (2) sont tentés de tricher pour connaître les préférences des autres utilisateurs et biaiser le résultat du calcul pour influencer la décision finale. On note ici que l'application est plus un vote qu'un sondage, puisqu'une décision finale est prise. Cependant le terme de vote électronique ayant pris, ces dernières années, une signification bien précise entraînant des contraintes strictes, on lui préfèrera le terme sondage : les applications visées dans ce travail tolèrent des imprécisions sur le résultat des calculs.

Les entrées d'un calcul dans un réseau social en ligne sont des informations personnelles, potentiellement sensibles, des utilisateurs qu'ils ne veulent pas voir diffusées. D'autre part, les utilisateurs attendent un résultat juste, ou au moins suffisamment précis. Enfin, comme souvent, certains utilisateurs tentent de tricher. La question de comment effectuer un calcul dans ces conditions et de manière extensible se pose rapidement. Dans l'état actuel des réseaux sociaux en ligne, les opérations collaboratives menées par les utilisateurs sont exécutées par une entité centrale (par exemple FacebookPoll [Kre10] et Doodle [Doo10]), à savoir le serveur de la compagnie orchestrant le réseau social en ligne. Tout d'abord, cette solution souffre clairement d'une faible extensibilité, c'est-à-dire qu'elle peine à supporter un très grand nombre d'utilisateurs. D'autre part, les compagnies orchestrant les réseaux sociaux, du fait de leur rapidité à s'attribuer puis utiliser les informations personnelles des utilisateurs à des fins commerciales, ne sont pas toujours considérées comme dignes de confiance. Pour ces raisons, les calculs dans les réseaux sociaux en ligne soulèvent des problèmes de confidentialité et appellent, pour éviter qu'une entité centrale ne collecte ces informations, à la répartition du calcul, c'est-à-dire à ce que les utilisateurs soient responsables de leurs informations personnelles et effectuent les calculs par eux-mêmes. On se trouve donc face à un problème

de calcul réparti décentralisé, extensible, et mettant en jeu des informations sensibles en présence d'utilisateurs malhonnêtes.

Un protocole d'agrégation simple effectuant des calculs intermédiaires en collectant les entrées de manière décentralisée, s'il résout le problème que pose une entité centrale, est extrêmement sensible aux utilisateurs malhonnêtes – ces derniers pouvant biaiser de manière arbitraire le résultat du calcul – et révèle les informations personnelles (les entrées du calcul) des utilisateurs (appelés nœuds) aux autres utilisateurs participant au calcul. Une méthode traditionnelle pour que les entrées du calcul ne soient pas connues consiste à diviser une entrée en plusieurs parts ne révélant seules pas ou peu d'informations sur l'entrée dont elles sont issues et telles que le résultat de l'agrégation des parts soit le même que le résultat de l'agrégation des entrées elles-mêmes. Cette technique est appelée *partage de secret par homomorphisme* [Ben86]. Concrètement, dans le cas où les entrées sont des entiers et la fonction calculée est la somme, les parts doivent vérifier que la somme des parts d'une entrée est égale à l'entrée. Cependant, un tel protocole n'empêche pas les utilisateurs malhonnêtes de biaiser le résultat en modifiant des résultats intermédiaires ou même en créant des parts invalides (c'est-à-dire ne correspondant pas à une entrée autorisée, par exemple 0 et 1 dans un sondage binaire).

Les protocoles de calcul multi-utilisateur sécurisé (MPC pour *secure multi-party computation*) [RBO89] qui utilisent intensivement la cryptographie permettent d'effectuer de manière exacte et privée des calculs répartis en présence d'une minorité d'utilisateurs potentiellement malhonnêtes. Les solutions basées sur MPC, y compris les plus récentes [DIK+08, DN07], bien que prometteuses restent peu extensibles (de complexité polynomiale en le nombre d'utilisateurs) et donc difficilement utilisables à grande échelle en pratique [DIK+08]. D'autre part, comme expliqué en introduction, une des motivations de ce travail est d'explorer les possibilités de fournir des garanties de sécurité sans cryptographie.

La thèse défendue dans ce chapitre est qu'un utilisateur malhonnête triche à moins que les risques d'être pris, et que la tricherie soit révélée aux membres du réseau social en ligne, ne soient élevés. En cela, le modèle d'utilisateurs considéré est plus contraint que celui d'utilisateurs byzantins [LSP82]. Il faut garder à l'esprit, et c'est la clé de voûte du travail présenté dans ce chapitre, que les informations relatives à un utilisateur du réseau social en ligne reflètent directement la personne réelle associée, supposée *respectable*. Peu d'utilisateurs se souciant de leur réputation souhaitent voir leur profil, accessible à tous et à leurs proches en particulier, annoté de la mention tricheur. On exploite cette caractéristique des entités d'un réseau social en ligne pour dissuader ces derniers de tricher, au lieu d'empêcher les tricheries à l'aide de techniques mathématiques complexes comme le font les protocoles cryptographiques ou de les dissimuler comme le font les protocoles tolérant les utilisateurs byzantins (BFT pour *Byzantine Fault-Tolerance*). En parallèle du protocole de calcul réparti lui-même, les nœuds effectuent des vérifications qui annotent le profil des tricheurs détectés en se basant sur les témoignages collectés. Si le fait que les utilisateurs disposent d'un profil public sert à les dissuader de tricher, le graphe des relations entre les utilisateurs permet de détecter, et lutter contre, les coalitions d'utilisateurs qui accusent à tort des utilisateurs honnêtes du système.

L'approche proposée dans ce chapitre repose sur le fait que chaque utilisateur est lié à une personne réelle et est, par conséquent, sensible à une attaque de Sybil (en référence au personnage principal du roman de Flora Rheta Schreiber souffrant de troubles dissociatifs de l'identité) [Dou02] où un utilisateur crée de fausses identités virtuelles pour l'aider à atteindre ses fins, c'est-à-dire biaiser le résultat et découvrir les informations personnelles des autres utilisateurs. Des travaux récents, exploitant les caractéristiques structurelles des réseaux sociaux en ligne, ont conduit à l'élaboration de

protocoles de détection de telles identités [TMLS09, YGKX08, YKGF08]. On peut également citer la politique de Battle.net [Bat], le réseau en ligne d'utilisateurs de l'éditeur de jeux Blizzard, qui force ses utilisateurs à s'identifier à l'aide de leur carte d'identité ou autre information personnelle. On ne considère dans ce chapitre que des utilisateurs réels et on exploite alors leur attachement à leur réputation pour les dissuader de tricher.

Pour illustrer le mécanisme d'annotation de profil, considérons une situation où une vérification recoupant les informations de Alice et de Bob démontre la tricherie de Charlie. Dans ce cas, le profil de Alice et de Bob sont annotés « Alice et Bob ont accusé conjointement Charlie » et le profil de Charlie sera annoté « Charlie a été accusé de tricherie par Alice et Bob ». Dans un réseau social en ligne, un utilisateur respectable ne souhaite pas voir son profil annoté comme celui de Charlie. Lorsque le protocole de vérification n'accuse pas à tort les utilisateurs honnêtes, le profil d'un nœud est annoté seulement si l'accusé est malhonnête ou si un sous-ensemble des accusateurs est malhonnête. Intuitivement, en supposant un réseau social en ligne dont la majorité des utilisateurs est honnête, les fausses accusations peuvent être distinguées des vraies et se retourner contre les accusateurs. Par exemple un utilisateur accusant continuellement les utilisateurs qu'il vérifie sera considéré comme suspect et ses accusations, à terme, perdront de la valeur. De la même manière, une accusation commune d'utilisateurs ayant une relation déclarée dans le réseau social (par exemple une coalition d'amis visant à salir la réputation d'un autre utilisateur), et c'est ici que rentre en compte le réseau, a moins de valeur qu'une accusation commune d'utilisateurs sans relation entre eux. Ceci est une conséquence de la faible densité des réseaux sociaux en ligne, c'est-à-dire que deux utilisateurs pris au hasard ont une très faible probabilité d'être liés. Ces méthodes de filtrage, décrites ici de manière informelle, sont accomplies naturellement par un utilisateur consultant un profil annoté. Elles peuvent également être implémentées de manière automatique comme le montrent les nombreux systèmes grand public déployés aujourd'hui, principalement dans les domaines des jeux en ligne massivement multi-joueur [GV08, KTCB05], des casinos en ligne [Pok], de la détection de courriers électroniques indésirables [MPGD08, SKY10] et des systèmes de recommandations [Dig]. L'approche généralement utilisée pour gérer le accusations faites par des utilisateurs d'un réseau social est la suivante : pour exclure un utilisateur il faut obtenir un consensus parmi un ensemble d'utilisateurs non liés. On notera ici que, contrairement à Facebook où les utilisateurs déclarent leurs relations, le réseau social n'est pas toujours explicite dans les applications considérées. Cependant, il peut généralement être inféré de manière efficace, en regardant quels utilisateurs ont l'habitude de jouer ensemble dans les cas des jeux en ligne par exemple. Dans ce chapitre, on considère que les utilisateurs malhonnêtes n'accusent pas à tort les utilisateurs honnêtes.

Pour récapituler, le système considéré dans ce chapitre est un réseau social en ligne composé d'utilisateurs honnêtes ou malhonnêtes correspondant à des personnes réelles. On note N le nombre total d'utilisateurs et B le nombre d'entre eux qui sont malhonnêtes. Les uns comme les autres se soucient de leur réputation et, par conséquent, ne veulent pas que leurs informations personnelles ou leurs tricheries, s'il trichent, soient révélées. Les utilisateurs du système sont modélisés de la manière suivante. Les utilisateurs honnêtes suivent scrupuleusement le protocole, y compris les vérifications. Les utilisateurs malhonnêtes en revanche s'autorisent à dévier du protocole et à mentir dans les procédures de vérification dès lors que ces déviations ne sont pas détectées à coup sûr, c'est-à-dire avec probabilité 1. Les utilisateurs malhonnêtes forment une coalition et peuvent donc collaborer pour arriver à leur fins. Enfin, pour les raisons évoquées ci-dessus, les utilisateurs malhonnêtes n'accusent pas à tort les utilisateurs honnêtes.

Les contributions de ce chapitre sont multiples. On présente tout d'abord DPol (pour *decentral-*

ized polling), un protocole réparti de sondage binaire (Section 4.2). En bref, DPol forme des groupes d'utilisateurs de taille $\sqrt{N}$ agissant comme des bureaux de vote, qui s'échangent leurs résultats partiels de telle sorte que tous les utilisateurs finissent par connaitre le résultat total du sondage. Les utilisateurs expriment leur opinion à l'aide d'une méthode de partage de secret par homomorphisme contrôlée par un paramètre de sécurité k. L'architecture utilisée par DPol le rend extensible (complexité en mémoire, en temps et en messages en $\mathcal{O}(\sqrt{Nk})$), garantit la confidentialité des utilisateurs (l'opinion d'un utilisateur est révélée avec une probabilité inférieure à $(B/N)^{k+1}$) et facilite les vérifications limitant ainsi l'impact des utilisateurs malhonnêtes (l'écart sur le résultat du sondage est d'au plus $(6k+2)B$) avec forte probabilité quand $B < N/2$ et N est grand. La borne sur l'impact des utilisateurs malhonnêtes s'applique de manière déterministe pour tout N si $B < \sqrt{N}$. Les résultats obtenus sont illustrés et confirmés à l'aide de simulations numériques. On rapporte les résultats du déploiement de DPol sur 400 ordinateurs de la plate-forme PlanetLab démontrant ainsi l'aspect pratique du protocole. À la lumière des défis et des contraintes rencontrées dans l'élaboration de DPol, on propose une formalisation du problème S^3, pour *Scalable and Secure computation in Social networks* : le calcul réparti d'un candidat, c'est-à-dire une fonction vérifiant certaines propriétés, dans un réseau social où les utilisateurs sont potentiellement malhonnêtes mais toujours soucieux de leur réputation. Un calcul S^3 garantit, dès lors que deux utilisateurs honnêtes ont des entrées différentes, que les utilisateurs malhonnêtes ne peuvent découvrir l'entrée d'aucun utilisateur honnête. On propose alors Agg-S^3, un protocole permettant de S^3 calculer une classe de fonctions d'agrégation incluant la somme composante-à-composante de vecteurs binaires ne comportant qu'un seul 1 (section 4.4). L'architecture et les méthodes utilisées par Agg-S^3 sont inspirées de DPol mais affinées de manière à garantir les propriétés spécifiées par le problème S^3 et à élargir la classe des fonctions calculées du sondage binaire aux fonctions k-Lipschitziennes. Les différences entre DPol et Agg-S^3 sont expliquées en détail à la fin de ce chapitre.

4.2 DPol : sondage binaire décentralisé

Cette section est consacrée au problème du sondage binaire, c'est-à-dire déterminer quelle option parmi deux est préférée par les utilisateurs. On considère un système de N nœuds identifiés de manière unique, dont B malhonnêtes (formant une coalition $\mathfrak{B}$). Chaque nœud p commence avec une entrée $v_p \in \{-1, +1\}$ (aussi appelée vote étant donné le contexte d'application) et le résultat attendu du sondage est la somme des entrées : $\sum_p v_p$. L'option favorite est déterminée simplement par le signe du résultat. Chaque nœud possède également un profil pouvant être annoté et le modèle de comportement des nœuds est celui présenté en introduction. On considère le scénario où les nœuds malhonnêtes veulent promouvoir la même option et on suppose, sans perte de généralité, qu'il s'agit de l'option représentée par la valeur -1. En d'autre termes, les nœuds malhonnêtes ont pour objectif, entre autres, de minimiser le résultat du sondage. On notera que le cas d'une unique coalition dont tous les nœuds partagent la même opinion représente un pire cas pour la sécurité du système.

Le protocole proposé repose sur une architecture précise (ou réseau couvrant, appelé *overlay* en anglais), indépendante du réseau social, permettant une dissémination efficace et privée de l'information tout en facilitant les vérifications réparties. Le réseau couvrant peut être mis en place par l'entité centrale orchestrant le réseau social ou de manière décentralisée. On notera, à titre d'exemple, que le protocole Kelips [GBL$^+$03] permet de construire un réseau couvrant similaire à celui utilisé par DPol de manière complètement décentralisée tout en assurant une répartition uni-

forme des nœuds malhonnêtes, et ce, malgré les attaques potentielles de ces derniers. Dans DPol, l'ensemble des nœuds est partitionné en r groupes ordonnés de g_0 à g_{r-1}. Un nœud p du groupe g_i maintient deux ensembles de nœuds avec lesquels il communique : l'ensemble $\mathcal{P}_o$ des nœuds appartenant au même groupe que lui, appelés ses *assesseurs* ($\mathcal{P}_o = g_i \backslash \{p\}$) et un ensemble $\mathcal{P}_p$ d'*intermédiaires* dans le groupe suivant ($\mathcal{P}_p \subseteq g_{i+1 \mod r}$). Par conséquent, les groupes forment virtuellement un anneau sur lequel g_{i+1} succède à g_i et g_0 à g_{r-1}. Chaque groupe est un sous-graphe complètement connecté. On appelle *client* d'un nœud p, un nœud ayant p comme intermédiaire. Chaque nœud maintient également l'ensemble $\mathcal{P}_c$ de ses clients ($\mathcal{P}_c \subseteq g_{i-1 \mod r}$). On suppose qu'un nœud ne communique qu'avec les nœuds dans un des trois ensembles qu'il maintient. Plus concrètement, les nœuds ignorent les messages provenant de nœuds en dehors de $\mathcal{P}_o \cup \mathcal{P}_c$. La figure 4.1 représente le réseau couvrant utilisé par DPol, centré autour d'un nœud p du groupe g_i.

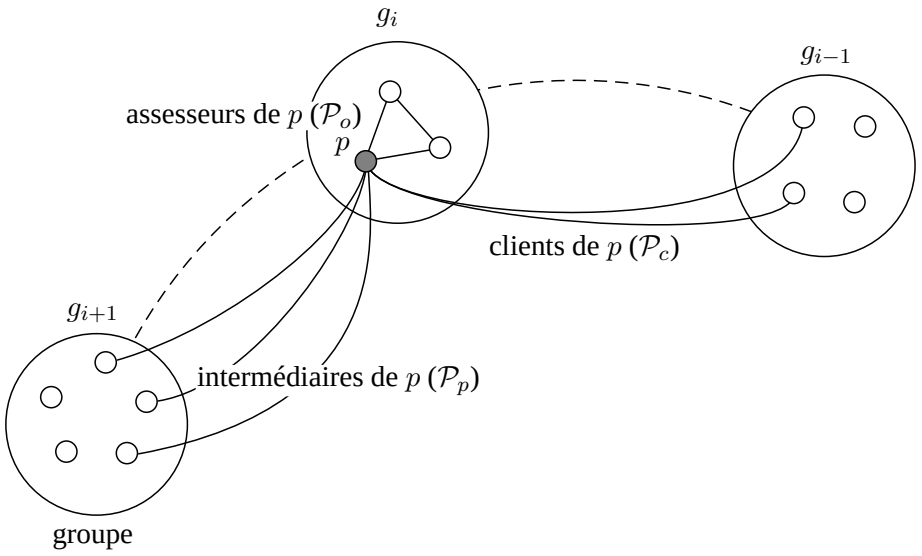


FIGURE 4.1 : Réseau couvrant utilisé par DPol

4.2.1 DPol : sondage binaire décentralisé

Cette sous-section décrit le protocole DPol, prouve sa correction et analyse ses performances en termes de complexité et de tolérance aux pertes de messages et pannes.

Description du protocole

DPol est composé de trois phases : (1) le vote, (2) le comptage intermédiaire et (3) la diffusion des scrutins locaux. En résumé, un nœud p (votant pour une des deux options $v_p \in \{-1, +1\}$) génère lors de la phase de vote un ensemble de bulletins, également à valeurs dans $\{-1, +1\}$, reflétant conjointement son vote (quand ils sont additionnés) et envoie chacun d'eux à un intermédiaire différent. Pendant la phase de comptage intermédiaire, les nœuds d'un groupe calculent conjointement la somme des votes des nœuds du groupe précédent, appelée scrutin local. Cette étape est réalisée en faisant en sorte que chaque nœud additionne les bulletins qu'il a reçus de ses clients et communique la somme ainsi obtenue à tous ses assesseurs. Enfin, les scrutins locaux ainsi calculés dans chaque

groupe sont diffusés aux membres des autres groupes le long de l'anneau en utilisant les même liens que durant la phase de vote, c'est-à-dire via les intermédiaires. Chaque nœud peut additionner les r scrutins locaux et ainsi calculer le résultat du sondage. Il ne reste alors qu'à déterminer l'option favorite en regardant le signe du résultat. Dans la suite de cette sous-section, on détaille chacune des trois phases de DPol.

Algorithme 1 Protocole DPol exécuté par un nœud p du groupe g_i, $i \in \{0, \ldots, r-1\}$

Entrées : un vote $v \in \{-1, +1\}$
Variables : scrutin individuel $t'' = 0$
scrutin local $t' = 0$
un tableau d'ensembles de scrutins locaux $\mathcal{S}[\{1, \ldots, r\} \to \varnothing]$
un tableau de scrutins locaux $\mathcal{T}[\{1, \ldots, r\} \to \perp]$
Sortie : résultat final $\hat{t}$

Algorithme de sondage

1: vote(v, $\mathcal{P}_p$)
2: comptage_local(t'', $\mathcal{P}_p$)
3: $t' = t' + t''$
4: diffusion_scrutin_local(i, t', $\mathcal{P}_p$)
5: $\hat{t} = \sum_i \mathcal{T}[i]$

Phase de vote

procédure vote(v,$\mathcal{P}_p$) **:**
6: $b = v$
7: **pour tout** $intermédiaire \in \mathcal{P}_p$ **faire**
8: envoyer [BULLETIN, b] $(intermédiaire)$
9: $b = -b$
10: **fin pour**

lors de l'événement ⟨ réception | [BULLETIN, b] ⟩ **faire**
11: $t'' = t'' + b$

Phase de comptage intermédiaire

procédure comptage_local(t'',$\mathcal{P}_o$) **:**
12: **pour tout** $assesseur \in \mathcal{P}_o$ **faire**
13: envoyer [SCRUTININDIVIDUEL, t''] $(assesseur)$
14: **fin pour**

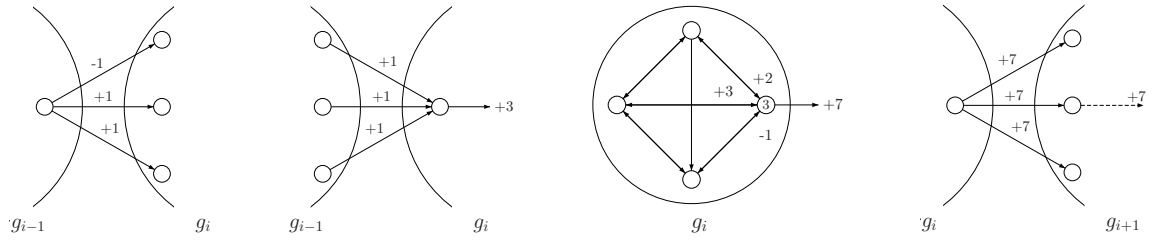
lors de l'événement ⟨ réception | [SCRUTININDIVIDUEL, t] ⟩ **faire**
15: $t' = t' + t$

Phase de diffusion des scrutins locaux

procédure diffuser_scrutin_local(i, t', $\mathcal{P}_p$) **:**
16: **pour tout** $intermédiaire \in \mathcal{P}_p$ **faire**
17: envoyer [SCRUTINLOCAL, i, t'] $(intermédiaire)$
18: **fin pour**

lors de l'événement ⟨ réception | [SCRUTINLOCAL, i_{groupe}, t] ⟩ **faire**
19: $\mathcal{S}[i_{groupe}] = \mathcal{S}[i_{groupe}] \cup \{t\}$
20: **si** ($\mathcal{S}[i_{groupe}] = |\mathcal{P}_c|$) **alors**
21: $\mathcal{T}[i_{groupe}] =$ choisir($\mathcal{S}[i_{groupe}]$)
22: diffuser_scrutin_local(i,$\mathcal{T}[i_{groupe}]$)
23: **fin si**

fonction choisir($\mathcal{A}$) **returns** scrutin local **:**
24: **renvoie** choisir le scrutin local le plus représenté $\mathcal{A}$



(a) Vote (b) Calcul du scrutin individuel (c) Calcul et envoi du scrutin local (d) Transmission des scrutins locaux

FIGURE 4.2 : Phases clés de DPol. Sur la figure 4.2a, un nœud de g_{i-1} génère 3 ($k = 1$) bulletins $\{+1, -1, +1\}$ et les envoie à ses intermédiaires dans g_i. La figure 4.2b montre un nœud de g_i collectant les bulletins $\{+1, +1, +1\}$ de ses clients, et les additionne pour obtenir $+3$ (son scrutin individuel) qu'il partage avec ses assesseurs dans g_i comme illustré sur la figure 4.2c. La figure 4.2c montre également que lorsqu'un nœud a reçu tous les scrutins individuels de ses assesseurs, ici $\{-1, +3, +2\}$, il en calcule la somme et envoie le scrutin local ainsi obtenu ($+7$) à ses intermédiaires dans le groupe suivant g_{i+1}. Enfin, on voit sur la figure 4.2d que les nœuds de g_{i+1} transmettent à leurs intermédiaires dans g_{i+2} le scrutin local de g_{i-1} obtenu de leurs clients dans g_i.

Phase de vote La méthode de génération des bulletins s'inspire d'un mécanisme de partage de secret proposé pour assurer la confidentialité des protocoles de population [DGFGR07]. Cette méthode partage également des points communs avec le mécanisme de Vote/Anti-vote/Vote [RS07]. Lorsqu'il veut voter pour une valeur v_p un nœud génère $k+1$ bulletins de valeur v_p et k bulletins de valeur $-v_p$ (lignes 6 à 10 dans l'algorithme 1). Le paramètre k est appelé *paramètre de sécurité* du protocole. Chacun des $2k+1$ bulletins générés est envoyé à un intermédiaire différent. Par conséquent, la taille de l'ensemble $\mathcal{P}_p$ doit être choisie en accord avec k, c'est-à-dire $|\mathcal{P}_p| = 2k+1$. La figure 4.2a représente un nœud votant $+1$ et envoyant ses bulletins à ses intermédiaires. Lorsqu'un nœud a reçu un bulletin de chacun de ses clients, la phase de vote est terminée.

Phase de comptage intermédiaire Un groupe g_i se comporte comme un bureau de vote pour les nœuds du groupe précédent sur l'anneau, c'est-à-dire g_{i-1}. Les nœuds dans g_{i-1} envoient leurs bulletins à leurs intermédiaires dans g_i. Pour calculer la somme des votes des nœuds de g_{i-1}, chaque nœud de g_i calcule tout d'abord la somme des bulletins qu'il a reçus (ligne 11 dans l'algorithme 1), appelée scrutin individuel t'', puis la transmet à tous ses assesseurs (lignes 12 à 14 dans l'algorithme 1). Un nœud de g_i additionne les scrutins individuels reçus de ses assesseur auxquels il ajoute le sien, obtenant ainsi un scrutin local t' égal à la somme des votes des nœuds de g_{i-1} (lignes 3 et 15 dans l'algorithme 1). La figure 4.2b illustre la phase de comptage intermédiaire.

Phase de diffusion des scrutins locaux Une fois qu'un nœud a calculé le scrutin local du groupe précédant le sien sur l'anneau, il commence la phase de diffusion de ce scrutin le long de l'anneau en le transmettant à ses intermédiaires (lignes 16 à 18 dans l'algorithme 1). Chaque nœud maintient un ensemble de valeurs possibles $\mathcal{S}[i]$ pour les scrutins locaux de chaque groupe $\{g_i\}_{i=0}^{r-1}$. Les valeurs des scrutins locaux utilisées pour le calcul du résultat final sont celles les plus représentées dans chacun de ses ensembles (ligne 21 dans l'algorithme 1). Ce mécanisme permet de se protéger des corruptions de scrutins et sera discuté plus en détail dans la section suivante. À la réception d'un scrutin local (ce scrutin ayant été émis ou transmis par l'un de ses clients), un nœud ajoute la valeur à l'ensemble des valeurs possibles pour le scrutin local du groupe correspondant (ligne 19 dans l'algorithme 1). Pour un groupe donné, un nœud détermine la valeur $\mathcal{T}[i]$ utilisée pour le scrutin local de ce groupe et la transmet lorsqu'il a reçu une valeur possible de chacun de ses clients (lignes 20 à 22 dans l'algorithme 1). Le résultat final est calculé en additionnant les valeurs de $\mathcal{T}$ quand celles-ci sont toutes disponibles : $\hat{t} = \sum_{i=0}^{r-1} \mathcal{T}[i]$. Cette phase est illustrée sur la figure 4.2d.

Analyse

On prouve maintenant la correction de DPol en supposant un environnement idéal, c'est-à-dire qu'il n'y a ni perte de messages ni panne ou tricherie des nœuds. Ces aspects sont considérés dans la suite en étudiant l'impact des pannes et pertes de messages de manières théorique et expérimentale (section 4.2.3), et en analysant l'impact des nœuds malhonnêtes dans le pire cas (section 4.2.2) et en moyenne (section 4.2.3).

Théorème 4.1 (Correction)**.** *Dans un système où chaque nœud a une entrée v_p, DPol termine et chaque nœud a calculé $\sum_p v_p$.*

Démonstration. (Exactitude) On prouve tout d'abord que les scrutins locaux calculés dans chacun des groupes correspondent effectivement à la somme des votes des nœuds du groupe précédent. Le

scrutin local calculé par un groupe est la somme des bulletins reçus par ses membres. Pour un nœud p, on note $\mathcal{B}_p = \{b_{p,1}, \ldots, b_{p,2k+1}\}$ les bulletins qu'il génère et envoie à ses $2k+1$ intermédiaires et on a : $\sum_{b \in \mathcal{B}_p} b = v_p$. De même, chaque nœud p' reçoit un ensemble $\mathcal{B}'_{p'}$ de bulletins de ses clients. Par conséquent, le scrutin individuel d'un nœud p' est $\sum_{b \in \mathcal{B}'_{p'}}$. Comme il n'y a ni perte de messages, ni panne, ni tricherie, l'ensemble des bulletins envoyés par les nœuds d'un groupe est égal à l'ensemble des bulletins reçus dans le groupe suivant, et donc le scrutin local calculé par g_i s'écrit :

$$t'_i = \sum_{p' \in g_i} \sum_{b \in \mathcal{B}'_{p'}} b = \sum_{p \in g_{i-1}} \sum_{b \in \mathcal{B}_p} b = \sum_{p \in g_i} v_p$$

On notera que ce résultat découle du fait que la méthode de partage de secret utilisée est un homomorphisme. Puisque les nœuds sont supposés honnêtes, les scrutins locaux sont transmis intacts, c'est-à-dire sans être corrompus, et le résultat final est donc la somme des votes des nœuds.

(Terminaison) Un nœud connaît le nombre exact de messages qu'il est supposé recevoir durant chaque phase. Étant donné que les nœuds envoient le bon nombre de messages (car ils sont honnêtes et non en panne) et que ces derniers sont effectivement reçus (car il n'y a pas de perte de messages), la fin de chaque phase peut être détectée par les nœuds et donc chaque phase se termine correctement. L'algorithme consistant en une séquence finie de phases, il se termine. □

Propriété 4.1 (Complexité spatiale)**.** *La taille asymptotique S de l'état d'un nœud du groupe g_i est $\mathcal{O}(r \cdot k \cdot |g_{i-1}| \,/\, |g_i| + |g_i|)$.*

Démonstration. Un nœud maintient l'ensemble de ses intermédiaires ($2k+1$), l'ensemble de ses assesseurs ($|g_i| - 1$) et l'ensemble de ses clients (en moyenne $(2k+1) \cdot |g_{i-1}| \,/\, |g_i|$). De plus, il doit stocker une valeur possible par client pour le scrutin local de chaque groupe (en moyenne $r \cdot (2k+1) \cdot |g_{i-1}| \,/\, |g_i|$), soit un total de $S = \mathcal{O}(k) + \mathcal{O}(|g_i|) + \mathcal{O}(k \cdot |g_{i-1}| \,/\, |g_i|) + \mathcal{O}(r \cdot k \cdot |g_{i-1}| \,/\, |g_i|) = \mathcal{O}(r \cdot k \cdot |g_{i-1}| \,/\, |g_i| + |g_i|)$. □

Propriété 4.2 (Complexité en messages)**.** *Le nombre asymptotique M de messages envoyés par un nœud du groupe g_i est $\mathcal{O}(r \cdot k + |g_i|)$.*

Démonstration. Un nœud envoie $2k+1$ messages durant la phase de vote (les bulletins qu'il envoie à ses intermédiaires), $|g_i| - 1$ messages durant la phase de comptage intermédiaire (son scrutin individuel qu'il envoie à ses assesseurs), et $r \cdot (2k+1)$ messages durant la phase de dissémination des scrutins locaux (pour transmettre chacun des scrutins locaux à ses intermédiaires). On obtient donc un total de $\mathcal{O}(r \cdot k + |g_i|)$. □

Dans un souci d'extensibilité, ces deux quantités doivent être minimisées. On note tout d'abord que les paramètres $\{|g_i|\}_{i=0}^{r-1}$ et r ne sont pas indépendants puisqu'ils sont liés par la relation suivante : $\sum_{i=0}^{r-1} |g_i| = N$. Par conséquent M et S sont minimisées pour $r = \sqrt{N/k}$ et $|g_i| = \sqrt{Nk}$ pour tout $i \in \{0, \ldots, r-1\}$ et on a : $S = M = \sqrt{Nk}$. On notera que l'utilisation d'une architecture partitionnant un système de N nœuds en $\sqrt{N}$ groupes de taille $\sqrt{N}$ fut proposée pour la première fois dans [GY87].

Propriété 4.3 (Complexité temporelle)**.** *Dans l'hypothèse d'un système à communications synchrones où le temps est découpé en intervalles, DPol s'exécute en $\mathcal{O}(\max_{i \in \{0,\ldots r-1\}} |g_i| + r \cdot k)$ intervalles de temps.*

Démonstration. La phase de vote s'exécute en $2k+1$ intervalles de temps car les nœuds du système envoient, en parallèle, chacun de leurs $2k+1$ bulletins successivement (au rythme d'un bulletin par intervalle de temps). Pour les mêmes raisons, la phase de comptage intermédiaire nécessite $\mathcal{O}(\max_{i\in\{0,\ldots,r-1\}} |g_i|)$ intervalles de temps. La diffusion d'un scrutin local d'un groupe au groupe suivant nécessite, tout comme la phase de vote, $2k+1$ intervalles de temps. L'anneau étant composé de r groupes, la complexité temporelle de DPol est donc $\mathcal{O}(\max_{i\in\{0,\ldots r-1\}} |g_i|+r\cdot k)$, soit $\mathcal{O}(\sqrt{Nk})$ en utilisant les valeurs issues de l'optimisation de S et M. □

4.2.2 Impact des nœuds malhonnêtes

On s'intéresse maintenant à l'étude de l'impact *maximum* des nœuds malhonnêtes sous l'hypothèse que leur comportement se limite aux déviations qui ne sont pas détectées à coup sûr. On suppose qu'une proportion α ($0.5 < \alpha < 1$) de nœuds vote pour l'option $+1$ et que les nœuds malhonnêtes promeuvent l'option -1 et veulent donc biaiser le résultat du sondage dans ce sens. Dans un souci de simplicité on suppose que le réseau couvrant est constitué de $\sqrt{N}$ (supposé entier) groupes de taille $\sqrt{N}$ et que chaque nœud a exactement $2k+1$ clients. Une fois les groupes construits, par exemple à l'aide de Kelips, on peut obtenir cette dernière propriété en ordonnant de manière aléatoire les nœuds de chaque groupe et en attribuant comme intermédiaires au nœud j du groupe g_{i-1}, les nœuds $j \mod \sqrt{N}, j+1 \mod \sqrt{N}, \ldots, j+2k \mod \sqrt{N}$ du groupe g_i.

DPol inclut des mécanismes répartis de vérification permettant de détecter les tricheries et identifier les nœuds malhonnêtes. On distingue deux types de vérifications : (1) les vérifications publiques, qui ne manipulent que des données non-confidentielles des utilisateurs, par exemple les scrutins individuels, et (2) les vérifications privées qui s'autorisent à utiliser le contenu des bulletins. On notera que les nœuds honnêtes participent toujours aux vérifications publiques mais peuvent refuser de participer aux vérifications privées. Comme expliqué en introduction, dans le cadre de vérifications les nœuds sont autorisés à annoter le profil des nœuds avec qui ils ont échangé des messages, incitant ainsi les nœuds malhonnêtes à ne pas tricher.

Vérifications publiques

Théorème 4.2 (Précision au pire cas)**.** *Chaque membre d'une coalition de $B < \sqrt{N}$ nœuds malhonnêtes peut impacter le résultat du sondage d'au plus $6k+2$.*

Structure. La preuve repose sur le fait que les nœuds honnêtes participent toujours aux vérifications publiques et que les nœuds malhonnêtes ne se comportent pas de telle sorte que leur tricherie est détectée à coup sûr. La méthodologie pour borner l'impact des nœuds malhonnêtes sur le résultat du sondage est la suivante : pour chaque tricherie leur permettant de biaiser le résultat du sondage, si elle est détectée à coup sûr par une vérification publique, alors ils ne l'implémentent pas. On exhibe donc, pour chaque tricherie, une procédure de vérification publique et on calcule l'impact maximal qu'un nœud malhonnête peut avoir sans être détecté. Les lemmes 4.1 à 4.4 couvrent toutes les tricheries possibles dans les différentes phases du protocole. On additionnant les impacts maximaux des différentes tricheries, on obtient un impact sur le résultat du sondage d'au plus $2k+2(2k+1) = 6k+2$. □

Lemme 4.1 (Vote)**.** *Pendant la phase de vote un nœud malhonnête peut biaiser le résultat du scrutin d'au plus $2k$.*

Démonstration. Comme expliqué dans la description du réseau couvrant, durant la phase de vote les nœuds ignorent les messages venant de nœuds qui ne sont pas leurs clients. Un nœud malhonnête ne peut donc envoyer qu'un total de $2k+1$ (ou moins) bulletins à ses intermédiaires. Ces bulletins sont à valeurs dans $\{-1,+1\}$ et leur somme est censée être dans $\{-1,+1\}$ également. Pour diminuer le résultat du sondage (on rappelle que les nœuds malhonnêtes promeuvent l'option représentée par -1), un nœud malhonnête peut remplacer les bulletins $+1$ qu'il est censé envoyer par des bulletins -1. Dans le pire cas, il envoie $2k+1$ bulletins -1, soit l'équivalent d'un vote de $-2k-1$. Son impact est alors l'écart minimal entre le résultat en cas de tricherie et un résultat valide, soit $\min\left(|-1-(-2k-1)|\,,|+1-(-2k-1)|\right)=2k$. □

Lemme 4.2 (Comptage intermédiaire : calcul du scrutin individuel)**.** *Il existe un mécanisme de détection publique tel qu'un nœud malhonnête modifiant son scrutin individuel de plus de* $2(2k+1)$ *est détecté à coup sûr.*

Démonstration. Le réseau couvrant utilisé par DPol assure que chaque nœud a $2k+1$ clients et reçoit donc $2k+1$ bulletins lors de la phase de vote. Un nœud malhonnête peut corrompre les bulletins reçus en changeant les bulletins $+1$ en bulletins -1. Il peut de manière générale, choisir une valeur arbitraire comme scrutin individuel. Cependant, les bulletins étant à valeurs dans $\{-1,+1\}$, la somme ne peut pas dépasser $2k+1$ en valeur absolue. Par conséquent, si les assesseurs d'un nœud vérifient que le scrutin transmis par ce dernier reste dans l'intervalle $[-2k-1;2k+1]$, l'impact d'un nœud malhonnête durant la phase de comptage est donc l'écart maximal entre une valeur possible du scrutin individuel et une valeur valide du point de vue de la vérification publique. Ce maximum est atteint lorsqu'un nœud malhonnête ne reçoit que des bulletins $+1$ et corrompt son scrutin individuel en $-(2k+1)$. L'impact maximal est donc de $2(2k+1)$. Il suffit qu'il y ait au moins un nœud honnête dans le groupe pour détecter, par une vérification publique, un scrutin individuel hors de cette borne. La borne sur le nombre de nœuds malhonnêtes $B<\sqrt{N}$ garantit cette propriété de sécurité. On notera également que, même si tous les nœuds d'un groupe sont malhonnêtes, ils ne peuvent pas, pour autant, biaiser le résultat de plus que $2(2k+1)$ par nœud malhonnête car on peut vérifier également de manière publique que le scrutin local calculé par un groupe est dans l'intervalle $[-(2k+1)\sqrt{N};(2k+1)\sqrt{N}]$. □

Lemme 4.3. *Il existe un mécanisme de vérification publique tel qu'un nœud envoyant des valeurs différentes à ses assesseurs lors de la diffusion des scrutins individuels dans son groupe est détecté à coup sûr.*

Démonstration. On considère le mécanisme suivant : lorsqu'un nœud a reçu les scrutins individuels de tous ses assesseurs, il transmet à ceux-ci l'ensemble des scrutins individuels qu'il a reçus. Ainsi, les nœuds honnêtes du groupe détectent les incohérences entre les valeurs du scrutin individuel qu'un nœud malhonnête leur a envoyées.

Il est important de noter ici, car ce n'est pas évident à première vue, qu'envoyer des valeurs différentes de scrutin individuel à ses assesseurs peut servir l'intérêt des nœuds malhonnêtes. En effet, étant donné qu'un nœud choisit comme valeur pour le scrutin local d'un groupe la valeur la plus fréquente parmi celles envoyées par ses clients, des clients malhonnêtes peuvent corrompre le scrutin local en obtenant la majorité. Pour ce faire un nœud malhonnête peut envoyer des valeurs de scrutin individuel différentes aux nœuds de son groupe conduisant ainsi à des valeurs distinctes de scrutin local et donc imposer facilement une valeur arbitraire comme scrutin local. Considérons le cas où $k=2$. Un nœud a donc 5 clients. Soit p un nœud dont deux clients sont malhonnêtes. Ces

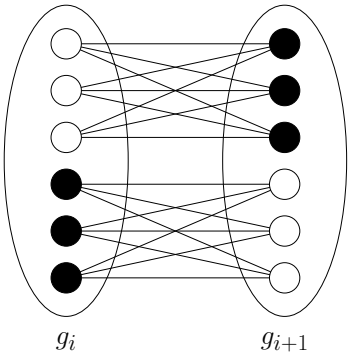


FIGURE 4.3 : Illustration de la preuve du lemme 4.4. Pour corrompre un scrutin local sans être détectée une coalition de nœuds malhonnêtes a besoin d'au moins $\sqrt{N}$ membres répartis dans deux groupes consécutifs. Les nœuds malhonnêtes sont représentés en noir.

derniers ne peuvent, a priori, pas obtenir une majorité lorsque p décide d'une valeur pour le scrutin local de leur groupe car les trois autres nœuds (honnêtes) envoient la même valeur. Cependant, si lors du comptage intermédiaire dans leur groupe les clients malhonnêtes envoient des valeurs différentes à ces trois nœuds, ces derniers enverront des valeurs différentes de scrutin local à p et les nœuds malhonnêtes obtiendront ainsi la majorité. □

Lemme 4.4. *Il existe un mécanisme de vérification publique qui détecte à coup sûr, pour $B < \sqrt{N}$, lorsque les nœuds d'un groupe n'envoient pas tous la même valeur pour un scrutin local et donc les nœuds malhonnêtes ne peuvent pas corrompre un scrutin local.*

Démonstration. On considère le mécanisme de vérification suivant : après avoir décidé de la valeur pour un scrutin local, un nœud transmet cette valeur à ses assesseurs. Une déviation est détectée par ce mécanisme si une des deux conditions suivantes n'est pas vérifiée (C1) un nœud honnête reçoit des valeurs différentes de ses clients ; (C2) la valeur choisie est différente de celle de ses assesseurs. Pour ne pas être détectés, des nœuds malhonnêtes envoyant un scrutin local corrompu à leur intermédiaires doivent s'assurer que tous les nœuds honnêtes dans le groupe suivant reçoivent la même valeur erronée. En effet, pour que (C1) soit vérifiée il faut que tous les clients d'un nœud honnête soient malhonnêtes (et s'accordent sur la valeur du scrutin corrompu). Pour que (C2) soit vérifiée, il faut que tous les nœuds honnêtes du groupe ait été induits en erreur. Si on considère un groupe contenant j nœuds malhonnêtes, il faut que les $\sqrt{N} - j$ nœuds honnêtes restants soient induits en erreur en respectant (C1). Un nœud ayant $2k+1$ clients et $2k+1$ intermédiaires, il faut au minimum $\sqrt{N}-j$ clients malhonnêtes pour induire en erreur $\sqrt{N}-j$ intermédiaires honnêtes. Soit un total de $j + \sqrt{N} - j = \sqrt{N}$ nœuds malhonnêtes, ce qui entre en contradiction avec l'hypothèse $B < \sqrt{N}$. Cette situation est illustrée sur la figure 4.3. En conclusion, une coalition de $B < \sqrt{N}$ nœuds malhonnêtes ne peut corrompre un scrutin local sans être détectée à coup sûr par un mécanisme de vérification publique, et donc son impact lors de la phase de diffusion est nul. □

On notera que la preuve repose sur l'hypothèse que le nombre de nœuds malhonnêtes est inférieur à $\sqrt{N}$ afin que les nœuds malhonnêtes ne puissent ni contrôler un groupe (c'est-à-dire qu'il y a au moins un nœud honnête par groupe), ni induire un groupe entier en erreur (c'est-à-dire qu'ils ne peuvent pas à la fois tricher et masquer leur tricheries) lors de la diffusion des

scrutins locaux. En réalité l'hypothèse minimale nécessaire ici est que deux groupes consécutifs sur l'anneau contiennent au plus $\sqrt{N}$ nœuds malhonnêtes, c'est-à-dire pour tout $i \in \{0, \ldots, r-1\}, |g_i \cap \mathfrak{B}| + |g_{i+1 \mod r} \cap \mathfrak{B}| < \sqrt{N}$. On étudie cette condition de manière probabiliste dans la sous-section 4.2.3 et on montre que la borne obtenue dans le théorème 4.2 reste valide avec une forte probabilité pour $B < N/2$ quand N tend vers l'infini.

Corollaire 4.1. *Si la marge de victoire de l'option favorite est supérieure à $(6k+2)B$, une coalition de $B < \sqrt{N}$ nœuds malhonnêtes ne peut pas changer la décision prise à l'issue du sondage.*

Démonstration. Ce résultat est une conséquence directe du théorème 4.2 dans la mesure où le résultat du sondage est la différence entre les scores des deux options, c'est-à-dire la marge. Si cette marge est inférieure à $(6k+2)B$, alors il se peut que les nœuds malhonnêtes, en trichant, puissent inverser le signe du résultat et donc la décision. □

On peut déduire de ce corolaire une relation entre la marge de victoire de l'option favorite et le nombre maximal de nœuds malhonnêtes que le système peut tolérer. La marge de victoire s'écrit en fonction de la proportion α de nœuds votant pour l'option $+1$: $N(2\alpha - 1)$. DPol peut donc tolérer jusqu'à $N(2\alpha-1)/(6k+2)$ nœuds malhonnêtes.

Vérifications privées

Jusqu'à présent on n'a envisagé que des mécanismes de vérification publique, c'est-à-dire ne manipulant que des scrutins et non des bulletins, de sorte que la confidentialité des entrées des nœuds ne soit pas compromise. On suppose désormais que les nœuds honnêtes sont prêts à sacrifier une partie de leur confidentialité pour détecter les tricheurs et donc obtenir une meilleure précision sur le résultat du sondage. On modélise la volonté des nœuds de révéler des informations privées par une probabilité non-nulle qu'un bulletin soit révélé par l'émetteur pour effectuer une vérification.

Théorème 4.3. *Un nœud malhonnête trichant durant la phase de vote ou de calcul intermédiaire est détecté avec une probabilité strictement positive (même si son impact sur le résultat est inférieur à $6k+2$).*

Démonstration. On considère, dans un premier temps, un nœud malhonnête p_1 qui envoie $k+1+j$ bulletins -1 et $k-j$ bulletins $+1$ ($1 \leq j \leq k$). Un mécanisme de vérification privée, demandant à p_1 de fournir une liste de j' intermédiaires à qui il a envoyé un bulletin -1 et de j' intermédiaires à qui il a envoyé un bulletin $+1$, détectera la tricherie de p_1 si j' est plus grand que $k-j$. On notera que, même pour $j' = k$, le vote de p_1 n'est pas dévoilé (à moins que le $(2k+1)$-ème bulletin ait été envoyé à un nœud malhonnête).

En ce qui concerne la phase de comptage intermédiaire, le scrutin individuel d'un nœud p_2 peut être borné plus précisément qu'avec une vérification publique en utilisant des informations fournies par ses clients. On suppose que n_b d'entre eux acceptent de communiquer la valeur du bulletin qu'ils ont envoyé à p_2 et que n_b^+ étaient des bulletins $+1$ et n_b^- étaient des bulletins -1. On peut alors réduire l'intervalle de vérification du scrutin individuel de $[-2k-1; 2k+1]$ (pour les vérifications publiques) à $[-2k-1+2n_b^+; 2k+1-2n_b^-]$.

Dans le cas des deux mécanismes de vérification privée envisagés, un nœud malhonnête en couvrant un autre s'expose lui-même aux vérifications privées. Par conséquent, l'hypothèse d'égoïsme des nœuds assure que le nœud malhonnête fautif est détecté. Considérons le cas où un nœud p_1

triche pendant la phase de vote, comme illustré sur la figure 4.4a, et que p_2 le couvre en témoignant avoir reçu un bulletin $+1$ de la part de p_1. Alors p_2 est exposé à une vérification privée sur son scrutin individuel. De même, dans la situation illustrée sur la figure 4.4b, si p_1 témoigne avoir envoyé un bulletin $+1$ à p_2 il s'expose lui-même à une vérification sur son vote. De manière générale, une tricherie commise par un nœud malhonnête de la coalition est forcément détectée si tous les nœuds honnêtes communiquent leurs bulletins lors de vérifications privées. Une expression de la probabilité de détecter un nœud malhonnête en fonction du nombre de bulletins corrompus est donnée dans la sous-section suivante. □

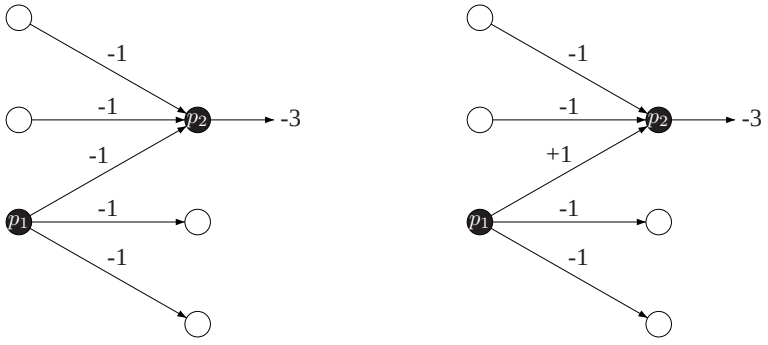


FIGURE 4.4 : Illustration de la preuve du théorème 4.3 : en couvrant un autre nœud, un nœud malhonnête s'expose lui-même aux vérifications privées.

4.2.3 DPol dans la pratique

L'analyse de DPol menée jusque-là considère les pires cas et a ainsi permis de borner l'impact des nœuds malhonnêtes. À titre d'exemple, dans le calcul de l'impact maximum durant la phase de comptage intermédiaire, on a considéré le cas où un nœud malhonnête promouvant l'option représentée par la valeur -1 ne recevait que des bulletins $+1$ lors de la phase de vote. Dans cette sous-section, on s'intéresse aux performances de DPol en pratique, c'est-à-dire au cas moyen. Les résultats présentés dans cette sous-section ont été obtenus soit par expérimentation soit par une analyse en moyenne probabiliste.

Le contexte expérimental consiste en un système de 400 nœuds de la plateforme PlanetLab qui se rapproche d'un déploiement grand public sur les critères pertinents ici : les pannes involontaires, les pertes de messages, et l'asynchronisme des communications. DPol a été implémenté en JAVA à l'aide de la librairie GenericRuntime. Durant les expériences, certains nœuds ont cessé de fonctionner. De plus les communications entre les nœuds sont effectuées via des connexions UDP sujettes aux pertes de messages. L'enchaînement des phases composant DPol ne peut donc pas être basé sur l'attente de *tous* les messages qu'un nœud est censé recevoir. Pour rendre DPol résistant aux pannes et aux pertes de messages, les phases sont bornées dans le temps et le choix des valeurs des scrutins locaux est fait après un temps fixe δt dès lors qu'un quorum de $\gamma \cdot \mathcal{P}_c$ nœuds est réuni. On a fixé γ à $0,5$ et δt à 5 secondes. Le réseau couvrant est construit de manière centralisée à partir de la liste complète des nœuds. Les $\sqrt{N}$ groupes sont formés aléatoirement et ont une taille moyenne de $\sqrt{N}$.

Chaque nœud a exactement $2k + 1$ intermédiaires choisis aléatoirement de manière uniforme dans le groupe suivant.

Confidentialité

La confidentialité se caractérise par la probabilité que le vote d'un nœud honnête soit découvert par une coalition de nœuds malhonnêtes.

Théorème 4.4 (Confidentialité)**.** *En supposant une proportion α, $0 < \alpha < 1$, de nœuds votant pour l'option représentée par $+1$, la probabilité, pour un nœud donné, que son vote soit découvert avec certitude par une coalition de nœuds malhonnêtes est bornée par $(B/N)^{k+1}$.*

Démonstration. Sans perte de généralité, on considère un nœud votant pour l'option représentée par $+1$. Son vote est découvert avec certitude si (1) le scrutin local de son groupe est égale à $(2k+1)\sqrt{N}$ car dans ce cas tous les nœuds du système savent avec certitude que tous les nœuds de ce groupe ont voté pour la même option ou (2) les $k + 1$ bulletins égaux à son vote sont connus des nœuds malhonnêtes. Le premier cas se produit avec une probabilité $\alpha^{\sqrt{N}-1}$. Le second cas signifie que les intermédiaires à qui les bulletins ont été envoyés sont malhonnêtes, ce qui se produit avec une probabilité $\binom{B}{k+1}/\binom{N}{k+1}$. La probabilité que le cas (1) se produise est négligeable par rapport à celle que (2) se produise pour N grand. En remarquant que $\binom{B}{k+1}/\binom{N}{k+1} \leq (B/N)^{k+1}$ pour tout k, B et N, on obtient le résultat voulu ce qui conclut la preuve. □

Précision

On évalue ici la précision du sondage, c'est-à-dire l'écart entre le résultat réel et le résultat obtenu à l'aide de DPol. On s'intéresse également au résultat de la décision finale basée sur le résultat, c'est-à-dire laquelle des deux options est considérée comme favorite. Pour générer les courbes présentées ici, on a fait varier la proportion de nœuds votant pour l'option $+1$ entre $0,5$ et 1. En l'absence de nœuds malhonnêtes le résultat du sondage en fonction de α est symétrique par rapport à $0,5$. En présence de nœuds malhonnêtes, seul cet intervalle est intéressant car ces derniers promeuvent l'option représentée par -1.

Pannes involontaires et pertes de messages La figure 4.5 représente la précision du résultat de DPol pour la valeur du paramètre de sécurité $k = 2$. Pour chaque valeur de α et pour l'ensemble des 25 expériences de Monte-Carlo, on évalue l'écart-type des résultats calculés par les nœuds. La figure 4.5a représente l'erreur moyenne en fonction de α. On notera que l'erreur commise est plus faible pour des valeurs de α proches de $0,5$. Cela est dû au fait que, dans ce cas, les proportions de bulletins $+1$ et de bulletins -1 sont égales et donc que les pertes de bulletins se compensent. D'autre part, les scrutins intermédiaires sont proches de zéro et donc leur perte n'a que peu d'impact sur le résultat final. Lorsque la barre d'erreur représentant l'écart-type est au-dessus de la droite horizontale représentant un résultat de zéro, cela signifie qu'une grande majorité des résultats des nœuds a le même signe que le résultat réel. La figure 4.5b représente la proportion de nœuds prenant la bonne décision (parmi ceux arrivant à un résultat), ce qui signifie que le signe du résultat qu'ils ont calculé est correct. Pour revenir à l'exemple de l'introduction, cela correspond au nombre de personnes qui amèneraient leurs enfants contre l'avis de la majorité des convives. On observe que pour une proportion α supérieure à $52,5\%$, c'est-à-dire une marge de victoire de 5%, la totalité des

nœuds prend la bonne décision. Comme expliqué plus tôt, à cause des pertes de messages et des pannes, certains nœuds ne peuvent décider d'une valeur pour un ou plusieurs scrutins locaux ou cessent simplement de fonctionner et n'aboutiront donc pas à un résultat final. On observe sur la figure 4.5b que ce phénomène affecte moins de 5% des nœuds et est indépendant de α.

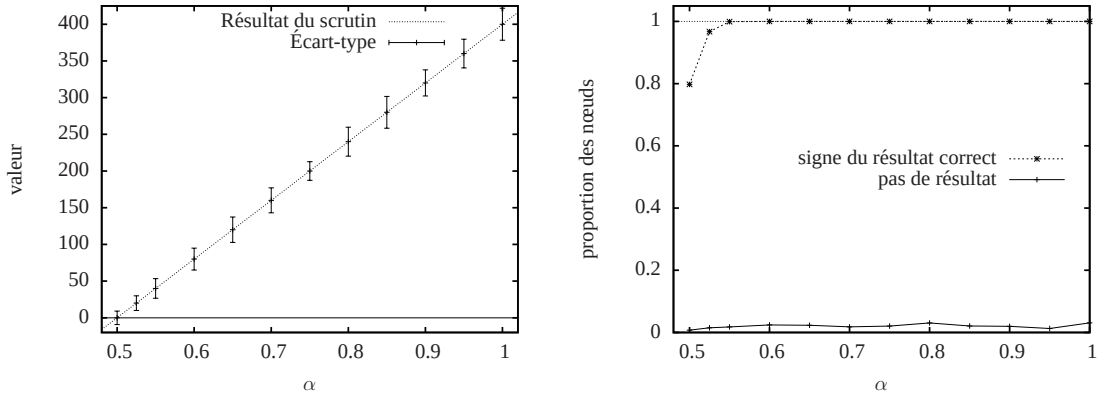


FIGURE 4.5 : Précision du résultat en présence de pannes involontaires et de pertes de messages ($N = 400$ and $k = 2$).

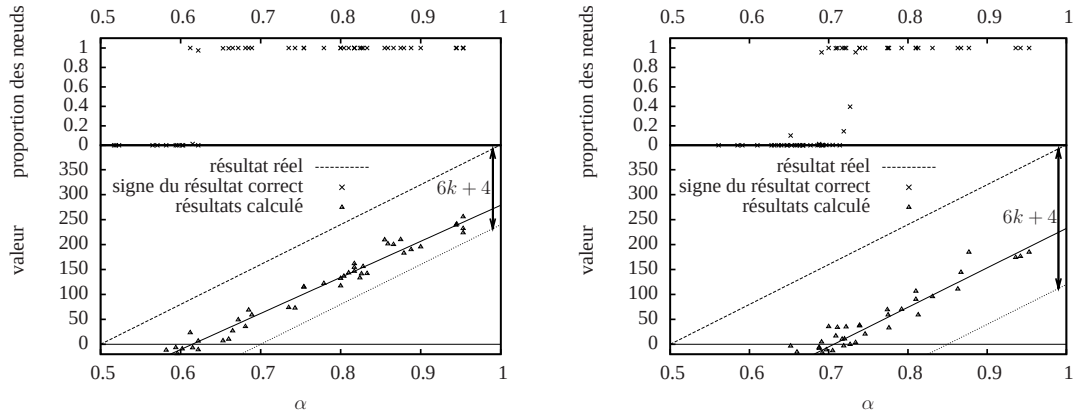
En suivant une démarche proche de la preuve du théorème 4.2, on peut borner l'impact des pertes de messages et des pannes. Là où l'impact d'une corruption était de 2 (en transformant un bulletin $+1$ en un bulletin -1) l'impact d'une perte est de 1 (de $+1$ ou -1 à 0). En conclusion, l'impact maximum des pertes sur le résultat final est de $3k + 2$. En moyenne, on peut également estimer l'impact moyen d'une perte de message en calculant l'espérance de la valeur du scrutin ou bulletin concerné. L'espérance de la valeur d'un bulletin est α et $\alpha \cdot (2k + 1)$ pour un scrutin individuel. Concernant les scrutins locaux, une perte impacte la proportion de nœuds ne pouvant pas décider d'une valeur car le quorum peut alors ne pas être réuni.

On note c la probabilité qu'un nœud tombe en panne de manière définitive et on calcule une expression de la probabilité qu'un nœud ne soit pas en mesure de décider d'une valeur pour au moins un scrutin local. On considère le scrutin local calculé dans le groupe 0 et on appelle $\bar{d}_i$ la probabilité qu'un nœud du groupe i ne parvienne pas à décider de la valeur de ce scrutin local. On a alors $\bar{d}_0 = 0$. Un nœud ne peut pas décider d'une valeur pour un scrutin local si moins de $\gamma \cdot (2k+1)$ de ses clients lui ont transmis une valeur pour ce dernier. On appelle $\bar{e}_i$ la probabilité que cela arrive pour un client appartenant au groupe i et on a alors : $\bar{e}_i = c + (1 - c)\bar{d}_i$, c'est-à-dire que ce dernier est en panne ou qu'il n'a lui-même pas pu décider. On obtient alors :

$$\bar{d}_i = \sum_{j=0}^{\gamma(2k+1)-1} \binom{2k+1}{j} (1 - \bar{e}_{i-1})^j \cdot \bar{e}_{i-1}^{2k+1-j} \quad . \tag{4.1}$$

En supposant que le système est synchrone et en remarquant que les scrutins locaux arrivant à un nœud transitent par la même suite de nœuds, on en déduit qu'un nœud du groupe r aboutit à un résultat si et seulement s'il a reçu le scrutin local calculé par le groupe 0. En conclusion, la probabilité d'aboutir à un résultat est $1 - \bar{d}_r$.

Tricheries (avec vérifications publiques) On introduit maintenant dans le système des nœuds malhonnêtes implémentant les attaques indétectables par les vérifications publiques : chaque nœud malhonnête envoie $2k+1$ bulletins -1 durant la phase de vote et transforme tous les bulletins $+1$ qu'il reçoit en bulletins -1. On exécute le protocole DPol pour plusieurs valeurs du paramètre de sécurité k dans un système de 400 nœuds dont 19 malhonnêtes ($B < \sqrt{N}$) et on représente, pour chacune des 25 expériences de Monte-Carlo, la proportion de nœuds obtenant un résultat du même signe que le résultat réel (points $\times$) ainsi que la moyenne des résultats obtenus par les nœuds (points $\triangle$). Les résultats sont représentés sur la figure 4.6.



(a) $k = 1$ (b) $k = 2$

FIGURE 4.6 : Précision du résultat en présence de $B = 19$ nœuds malhonnêtes sur un total de 400 nœuds : la coalition arrive à forcer la majorité des nœuds à décider, à tort, que l'option -1 est la favorite quand (a) $\alpha < 0.62$ pour $k = 1$, et quand (b) $\alpha < 0.73$ pour $k = 2$.

On remarque tout d'abord que l'impact des nœuds malhonnêtes est bien inférieur à la borne théorique calculée dans le théorème 4.2 (représentée par une ligne discontinue, le résultat réel étant représenté par une ligne pointillée). L'impact d'un nœud malhonnête dépend, en effet, du nombre de bulletins qu'il peut inverser et donc de la proportion α de nœuds préférant l'option représentée par $+1$. L'impact moyen peut être calculé facilement et vérifié expérimentalement. Durant la phase de vote, l'impact d'un nœud malhonnête est de $2k$, quel que soit α. Durant la phase de comptage intermédiaire, l'impact est égal à deux fois le nombre de bulletins $+1$ reçus, soit $\alpha\cdot(k+1)+(1-\alpha)\cdot k$ car les nœuds votant pour $+1$ envoient $k+1$ bulletins $+1$ et les nœuds votant -1 n'en envoient que k. On en déduit que l'impact total moyen est de $2k + 2(\alpha \cdot (k+1) + (1-\alpha) \cdot k) = 4k + 2\alpha$. Le résultat attendu en présence de B nœuds malhonnêtes s'écrit donc :

$$(2\alpha - 1) \cdot N - B(4k + 2\alpha) = 2(N - B)\left(\alpha - \frac{1}{2}\right) - (4k+1) \cdot B$$

On réalise une interpolation linéaire du nuage de 55 points obtenus avec $k = 2$ (ligne pleine dans la figure) et on obtient une pente de 791 pour une ordonnée à l'origine de -163, ce qui est proche des valeurs théoriques ($2(N - B) = 760$ et $-B \cdot (4k + 1) = -180$). On peut projeter ces résultats à plus grande échelle. Pour un système de 10.000 nœuds avec $B = \sqrt{N} - 1$ nœuds malhonnêtes et $k = 1$, la valeur minimale de α pour laquelle une majorité des nœuds prennent la bonne décision est $0,52$.

Tricheries (avec vérifications privées) On a montré dans la sous-section précédente qu'il était possible de détecter, avec une probabilité non-nulle, les tricheurs dont les déviations ne pouvaient être détectées par les vérifications publiques. On a également montré qu'un nœud malhonnête en couvrant un autre lors de vérifications privées s'exposait lui-même, ce qui a permis, grâce à l'hypothèse d'égoïsme, d'écarter ce comportement. On étudie donc ici, en supposant que les nœuds malhonnêtes ne se couvrent pas, la probabilité de détecter un nœud malhonnête à l'aide de vérifications privées, en fonction de l'ampleur de sa tricherie et de la probabilité p_d qu'un nœud honnête dévoile un bulletin donné.

On considère un nœud malhonnête qui envoie, durant la phase de vote, $k' > k+1$ bulletins -1. Ce nœud est détecté par une vérification privée si, au moins, $k+2$ nœuds ayant reçu un bulletin -1 de p acceptent de révéler ce bulletin lors des vérifications. Un nœud révèle le bulletin reçu de p avec une probabilité p_d sauf s'il est malhonnête, auquel cas il refuse. La probabilité que p soit détecté par une vérification privée est donc :

$$\sum_{j=k+2}^{k'} \binom{k'}{j} {p'_d}^j (1-p'_d)^{k'-j} \ ,$$

où $p'_d = (1 - B/N) \cdot p_d$.

De la même manière, on peut exprimer la probabilité qu'un nœud malhonnête p corrompant, lors de la phase de comptage intermédiaire, des bulletins reçus lors de la phase de vote soit détecté par une vérification privée. On suppose que p a reçu n_b^+ bulletins $+1$ et qu'il en corrompt n'. Dans ce cas, la probabilité de détection s'écrit :

$$\sum_{j=n_b^+-n'+1}^{n_b^+} \binom{n_b^+}{j} {p'_d}^j (1-p'_d)^{n_b^+-j} \ .$$

Sécurité

La preuve de la borne du théorème 4.2 sur l'impact maximal des nœuds malhonnêtes repose sur le fait que leur nombre dans deux groupes consécutifs est inférieur à la taille d'un groupe. L'hypothèse $B < \sqrt{N}$ garantit cette propriété de manière déterministe. Cependant, en considérant une répartition aléatoire uniforme des nœuds malhonnêtes, on montre que cette propriété est garantie avec une forte probabilité même au-delà de $\sqrt{N}$ nœuds malhonnêtes.

Théorème 4.5 (Sécurité probabiliste)**.** *La probabilité que toutes les paires de groupes consécutifs contiennent moins de $\sqrt{N}$ nœuds malhonnêtes est égale à 1 pour $B \geq N/2$, et converge exponentiellement vite en $\sqrt{N}$ vers 0 quand $B < N/2$.*

Démonstration. On dit qu'une paire $\{g_i, g_{i+1}\}$ de groupes consécutifs est compromise si $|g_i \cap \mathcal{B}| + |g_{i+1 \mod \sqrt{N}} \cap \mathcal{B}| \geq \sqrt{N}$. Comme g_i et $g_{i+1 \mod \sqrt{N}}$ sont disjoints, cette inégalité est équivalente à $|G_i \cap \mathcal{B}| \geq \sqrt{N}$ où $G_i = g_i \cup g_{i+1 \mod \sqrt{N}}$ pour tout $i \in \{0, \ldots, \sqrt{N}-1\}$.

On montre tout d'abord la première partie du théorème. On procède par contraposition en supposant que $|g_i \cap \mathcal{B}| + |g_{i+1 \mod r} \cap \mathcal{B}| < \sqrt{N}$ pour tout i. En sommant les $\sqrt{N}$ inégalités pour $i \in \{0, \ldots, \sqrt{N}-1\}$, on obtient que $\sum_{i=0}^{\sqrt{N}-1} |g_i \cap \mathcal{B}| + |g_{i+1} \cap \mathcal{B}| < N$, c'est-à-dire,

$2\sum_i |g_i \cap \mathcal{B}| < N$ puisque les $\{g_i\}_{i\in\{0,\ldots,\sqrt{N}-1\}}$ sont disjoints deux à deux. On en déduit que $2B < N$ et donc que $B < N/2$ ce qui conclut la première partie de la preuve.

On considère maintenant le cas où $B < N/2$ et on définit $\beta = B/N < 1/2$ comme la proportion de nœuds malhonnêtes dans le système. En utilisant l'inégalité de Hoeffding pour un tirage sans remise dans une population finie, on obtient que la probabilité qu'une paire de groupes consécutifs soit compromise est :

$$\begin{aligned}\forall i \in \{0,\ldots,\sqrt{N}-1\},\ \mathbb{P}\left[|G_i \cap \mathcal{B}| \geq \sqrt{N}\right] &= \mathbb{P}\left[|G_i \cap \mathcal{B}| - 2\sqrt{N}\beta \geq 2\sqrt{N}\left(\frac{1}{2}-\beta\right)\right] \\ &\leq \exp\left(-4\sqrt{N}\left(\frac{1}{2}-\beta\right)^2\right) \ .\end{aligned}$$

Le membre de droite converge exponentiellement vite en $\sqrt{N}$ car $\beta < 1/2$. En utilisant l'inégalité de Boole, on obtient que la probabilité qu'au moins une paire de groupes consécutifs soit compromise est bornée comme suit :

$$\begin{aligned}\mathbb{P}\left[\bigcup_i \{|G_i \cap \mathcal{B}| \geq \sqrt{N}\}\right] &\leq \sum_{i=1}^{\sqrt{N}} \mathbb{P}\left[|G_i \cap \mathcal{B}| \geq \sqrt{N}\right] \\ &\leq \sqrt{N}\exp\left(-\sqrt{N}\left(1-2\beta\right)^2\right) \ ,\end{aligned}$$

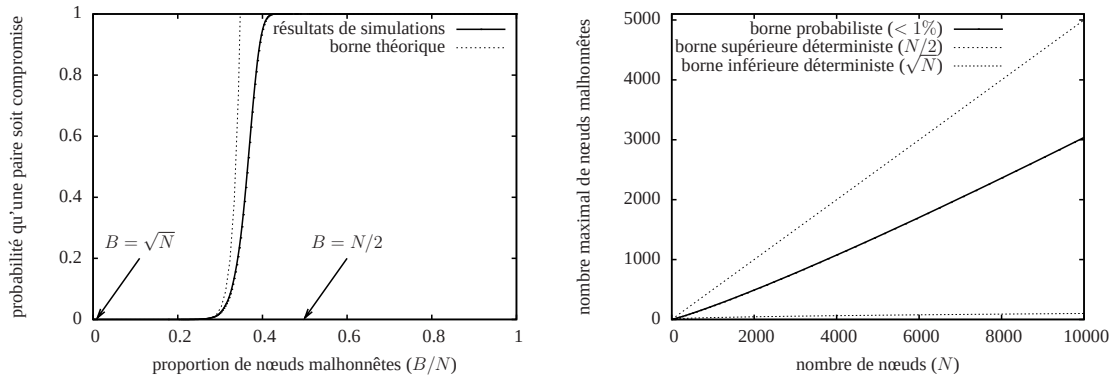
ce qui prouve une convergence exponentielle vers 0 quand N tend vers l'infini lorsque $B < N/2$.

On en conclut qu'une transition de phase exponentielle se produit quand la moitié des nœuds du système est malhonnête et, par conséquent, la borne de $(6k+2)B$ sur l'erreur du résultat de DPol est vérifiée en réalité tant que moins de 50% des nœuds sont malhonnêtes. □

On rapporte les résultats de simulation confirmant l'étude théorique de la sécurité de DPol. On construit les groupes de manière aléatoire en choisissant successivement $\sqrt{N}$ nœuds de manière uniforme et on mesure la proportion de réseaux couvrants où au moins une paire de groupes consécutifs est compromise. Sur la figure 4.7a, on trace cette probabilité en fonction de B dans un système de $N = 10.000$ nœuds. Les résultats obtenus sont proches de la borne théorique calculée dans le paragraphe précédent. On remarque que pour $B < 3.000$, cette probabilité reste inférieure à 1%. Cette probabilité n'atteint 0 que pour $B < 100$. On constate donc qu'en acceptant une probabilité de 1% qu'au moins une paire de groupes consécutifs soit compromise, le nombre de nœuds malhonnêtes tolérés est très largement supérieur. On trace donc, sur la figure 4.7b, le nombre maximum de nœuds malhonnêtes tolérés à 1% en fonction de N.

4.2.4 Résumé

Dans cette sous-section, on a présenté un protocole réparti de sondage binaire. En s'appuyant sur une structure de groupes permettant une collecte extensible d'information et facilitant les vérifications on a obtenu des garanties de précision et de confidentialité en présence de nœuds malhonnêtes liés à un réseau social. Cependant certains points restent à éclaircir, ou plutôt à définir formellement. On peut par exemple se demander ce qu'est un protocole extensible. Pour la précision, la métrique



(a) (b)

FIGURE 4.7 : Sécurité probabiliste de DPol.

utilisée dans l'analyse de DPol était la marge de sécurité du sondage, c'est-à-dire à partir de quelle proportion de nœuds votant pour l'option majoritaire le résultat est-il correct. Cette métrique est d'une part très liée à l'application : le sondage, et d'autre part elle ne spécifie pas ce qu'est un protocole précis. Est-ce que fournir un résultat correct pour $\alpha > 55\%$ est acceptable ? Enfin, la propriété de confidentialité de DPol repose sur une hypothèse forte : le fait que l'entrée d'un nœud est une variable aléatoire et que chaque valeur est choisie avec une probabilité non-nulle. Est-ce réaliste ? La confidentialité ne devrait-elle pas être indépendante de la manière dont les nœuds choisissent leur entrée ? La sous-section suivante étend la problématique du sondage à celle du calcul d'une fonction quelconque et répond aux questions posées ici en définissant de manière formelle les propriétés clés d'un calcul réparti dans un réseau social.

4.3 Le problème S^3

Le travail de conception d'un protocole de sondage binaire décentralisé a fait apparaître les contraintes et mesures de performances relatives au calcul réparti dans un réseau social. Ces dernières peuvent être résumées en deux mots : extensibilité et sécurité, sécurité faisant référence ici à la précision (contrainte classique pour mesurer la qualité d'un calcul) et à la confidentialité des entrées des entités (contrainte incontournable dans le contexte des réseaux sociaux où la valeur d'entrée d'une entité correspond en fait à des informations privées relatives à la personne réelle associée à cette entité). Cette section définit et formalise le problème du calcul reparti sécurisé et extensible dans les réseaux sociaux : le problème S^3 (pour *Scalable and Secure in Social networks*).

4.3.1 Candidat

Le problème S^3 fait tout d'abord intervenir un ensemble de nœuds $\Pi = \{p_1, \ldots, p_N\}$ et un candidat à S^3, c'est-à-dire une fonction à calculer.

Définition 4.1 (Candidat à S^3)**.** *Un candidat à S^3 est un quadruplet (f, V, U, d), où V est un ensemble quelconque, f est une fonction qui, à une séquence de taille quelconque d'éléments de V,*

associe un élément de U, $f : V^* \to U$ telle que $f(v_1, \ldots, v_N) = f(v_{\sigma(1)}, \ldots, v_{\sigma(N)})$ pour toute permutation σ de ses entrées, et (U, d) est un espace métrique.

Chaque nœud de Π possède une valeur initiale dans V appelée entrée, et un candidat à S^3 associe les entrées des nœuds du système à une valeur dans un espace métrique. La fonction f d'un candidat à S^3 est supposée symétrique vis-à-vis des entrées des nœuds, par conséquent elle peut être vue comme une fonction d'un multi-ensemble de valeurs de V car elle ne dépend que du multi-ensemble des entrées mais pas de leur assignation aux différents nœuds. Pour illustrer cette définition, on peut modéliser le problème du sondage binaire traité dans la section précédente par le candidat à S^3 suivant : $((v_1, \ldots, v_N) \mapsto v_1 + \cdots + v_N, \{-1, +1\}, \mathbb{Z}, (z_1, z_2) \mapsto |z_1 - z_2|)$. On pourra également considérer l'addition composante-à-composante de vecteurs d'entiers de taille k ($U = \mathbb{Z}^k$), où V est l'ensemble des vecteurs dont toutes les composantes sont nulles sauf une dont la valeur est soit $+1$ soit -1. Dans ce cas, la distance serait la distance de Manhattan correspondant à la norme ℓ^1.

Un nœud du système considéré dans le problème S^3 est lié à un utilisateur d'un réseau social par l'intermédiaire d'un profil public. À ce titre, les nœuds peuvent (1) communiquer à l'aide de messages privés et (2) annoter le profil des autres nœuds. À la fin du protocole, chaque nœud p a calculé une valeur o_p dans U (le résultat estimé du candidat à S^3) et un ensemble $\mathcal{T}_p$ de nœuds détectés comme étant des tricheurs. Cette dernière information est finalement diffusée sur le profil des nœuds impliqués. Ce mécanisme d'annotation combiné au modèle d'utilisateurs présenté dans les sections précédentes réduit les possibilités de déviation des tricheurs.

On définit un calcul réparti $\mathcal{D}$ conduit par un ensemble de nœuds Π comme une séquence de messages échangés et de calculs locaux effectués par les nœuds de telle sorte que chaque nœud honnête p aboutisse à un résultat, c'est-à-dire un élément o_p de U. On suppose que les choix locaux effectués par les nœuds sont aléatoires, par conséquent les $\{o_p\}_{p \in \Pi}$ sont des variables aléatoires. Dans la suite, on définit les propriétés souhaitées pour un calcul réparti dans un réseau social, à savoir l'extensibilité et la sécurité.

4.3.2 Extensibilité

L'extensibilité caractérise la capacité d'un système ou d'un protocole à gérer un grand nombre de participants. Dans le cas d'un calcul réparti, cela signifie que le coût en ressources du protocole grandit raisonnablement avec le nombre d'entrées N. Par conséquent, les propriétés des protocoles considérés sont exprimées sous forme de notations asymptotiques.

Définition 4.2 (h-Extensibilité)**.** *Un calcul réparti $\mathcal{D}$ est dit h-extensible si la complexité en termes de messages, d'espace mémoire et de nombre d'opérations pour chaque nœud est $\mathcal{O}(h(N) \cdot \operatorname{polylog} N)$.*

Le facteur poly-logarithmique est dû au fait qu'il faut $\log_2 N$ bits pour représenter l'identifiant d'un nœud. Par conséquent, comparer deux identifiants coûte $\log_2 N$ opérations élémentaires. On considère donc qu'un protocole est h-extensible s'il utilise $\mathcal{O}(h(N))$ unités de ressources par nœud où une unité correspond à une opération sur (ou une zone mémoire pour stocker) un identifiant, soit un nombre poly-logarithmique d'opérations élémentaires.

4.3.3 Précision

La définition de la précision repose sur la structure de l'espace métrique de l'ensemble d'arrivée du candidat à S^3 car elle permet de quantifier à l'aide d'un nombre réel l'écart entre le résultat du calcul réparti et le véritable résultat, c'est-à-dire $f(v_1, \ldots, v_N)$. Pour être significatif cet écart est normalisé par la distance maximale entre deux véritables résultats possibles, c'est-à-dire le diamètre de $f(V^N)$ (l'image par f de l'ensemble des séquences de N éléments de V).

Définition 4.3 (h-Précision)**.** *Un calcul réparti $\mathcal{D}$ calcule de manière h-précise un candidat (f, U, V, d) à S^3 si :*

$$\frac{1}{\Delta(N)} \cdot \max_{p \text{ honnête}} d(o_p, f(v_1, \ldots, v_N)) = \mathcal{O}\left(\frac{1}{h(N)}\right) \ ,$$

où v_i est l'entrée du i-ème nœud et

$$\Delta(N) = \max_{\substack{x_1, \ldots, x_N \\ y_1, \ldots, y_N}} d(f(x_1, \ldots, x_N), f(y_1, \ldots, y_N)) \ .$$

Cette définition illustre l'importance du choix de la distance sur l'ensemble d'arrivée U. Concrètement, tout ensemble peut être muni d'une distance grossière $d(x, y) = 0$ si $x = y$ et 1 sinon. Mais dans ce cas, seuls les protocoles répartis calculant exactement le véritable résultat résolvent le problème S^3 du point de vue de la précision (pour une fonction h qui tend vers l'infini quand N tend vers l'infini). Dans le contexte du sondage binaire par exemple, en considérant la distance naturelle sur les entiers relatifs (c'est-à-dire $d(x, y) = |x - y|$), les protocoles qui satisfont la h-précision sont ceux qui commettent une erreur inférieure à $N/h(N)$ (car $\Delta(N) = 2N$).

4.3.4 Confidentialité

La confidentialité se caractérise par la quantité d'information sur les entrées des nœuds révélée par le protocole aux nœuds participant au calcul. Plus précisément, elle décrit à quel point les informations acquises par une coalition de nœuds malhonnêtes au cours du calcul réparti permettent aux membres de cette dernière de découvrir le vote d'un nœud honnête. L'information obtenue par une telle coalition provient (1) des entrées de ses membres, (2) des messages reçus par ses membres au cours du calcul (par les échanges de messages spécifiés par le protocole) et (3) du résultat du calcul. Seul le point (2) est imputable au protocole de calcul réparti. Considérons le cas où tous les nœuds honnêtes ont la même entrée, v_0 par exemple, et que cette configuration d'entrées soit la seule qui, combinée aux entrées des nœuds malhonnêtes, conduise au résultat obtenu. Dans ce cas, les nœuds malhonnêtes savent pour sûr que tous les nœuds honnêtes ont v_0 pour entrée. Plus concrètement dans le cas du sondage binaire, le résultat est la somme des entrées de nœuds ($+1$ ou -1). Le résultat du protocole moins la somme des entrées des nœuds malhonnêtes donne la somme des entrées des nœuds honnêtes. Si cette somme est égale au nombre de nœuds honnêtes par exemple, c'est-à-dire qu'ils ont tous voté $+1$, alors les nœuds malhonnêtes savent avec certitude le vote de tous les nœuds honnêtes. Par conséquent ces situations doivent être écartées dans l'analyse de la confidentialité. On appelle une telle situation une configuration non-ambiguë d'entrées et on la définit formellement de la manière suivante :

Définition 4.4 (Configuration non-ambiguë d'entrées)**.** *Une configuration d'entrées v dans V^* est dite non-ambiguë du point de vue d'une coalition B de nœuds malhonnêtes s'il existe un nœud $p \notin B$ tel que, pour toute configuration d'entrées v' qui coïncide avec v pour tous les nœuds de B, $f(v) = f(v')$ implique $v_p = v'_p$.*

Définition 4.5 (Configuration triviale d'entrées)**.** *Une configuration d'entrées v dans V^* est dite triviale du point de vue d'une coalition B de nœuds malhonnêtes si tous les nœuds honnêtes ont la même valeur d'entrée.*

Un candidat à S^3 étant, par définition, symétrique par rapport aux entrées des nœuds, une configuration où au moins deux nœuds honnêtes ont des entrées différentes (appelée configuration non-triviale) est non-ambiguë puisqu'en permutant leurs entrées le résultat reste le même. On notera cependant qu'une configuration triviale peut être ambiguë. Lorsque la fonction calculée est la somme par exemple et que l'ensemble de valeurs possibles pour les entrées est $V = \{1, 2, 3\}$ la configuration d'entrée où tous les nœuds honnêtes ont pour entrée 2 donne le même résultat que la configuration où la moitié d'entre eux a pour entrée 1 et l'autre moitié a 3.

On considèrera dans la suite deux propriétés de confidentialité : la confidentialité *forte* et la confidentialité *faible*. Ces deux propriétés sont identiques sauf que dans la première seules les configurations d'entrée non-ambiguës sont écartées alors que dans la seconde toutes les configurations triviales sont écartées. La seconde est donc plus faible que la première puisqu'elle écarte un surensemble des configurations d'entrée écartées par la première. Dans le cadre du problème S^3 où les fonctions considérées sont symétriques, la confidentialité forte implique la confidentialité faible. On notera également que dans le cas du sondage binaire, les configurations ambiguës sont les configurations non-triviales et les deux propriétés de confidentialité sont donc équivalentes. En effet, dans le cas du sondage binaire une configuration triviale est toujours non-ambiguë car la seule configuration d'entrées donnant un total de $N - |B|$ (respectivement $-(N - |B|)$) est celle où tous les nœuds honnêtes ont $+1$ (respectivement -1) pour entrée.

Dans le contexte du problème S^3, un calcul réparti est dit privé (ou qui conserve la confidentialité) si la probabilité de découvrir avec certitude l'entrée d'au moins un nœud honnête – en supposant que la configuration d'entrées est ambiguë ou non-triviale – est $\mathcal{O}(N^\alpha)$ pour un certain $\alpha > 0$. Une telle garantie probabiliste est couramment utilisée et notée *avec forte probabilité* (et abrégée w.h.p pour *with high probability*). On définit formellement la propriété d'anonymat probabiliste en se basant sur la notion de trace de messages :

Définition 4.6 (Trace de messages)**.** *Une* trace de messages *(ou simplement* trace*) d'un calcul réparti est l'ensemble des messages échangés lors de l'exécution du calcul. Une trace est dite* compatible *avec une configuration d'entrées v si celle-ci peut être obtenue avec une probabilité non-nulle (on rappelle qu'une trace est une variable aléatoire) à partir de la configuration v. Deux traces sont dites équivalentes du point de vue d'une coalition de nœuds malhonnêtes B si elles sont obtenues avec la même probabilité et si tous les nœuds de B reçoivent exactement les mêmes messages dans les deux exécutions.*

On est maintenant prêts à définir la notion d'anonymat probabiliste qui décrit les garanties de respect de vie privée souhaitées dans le problème S^3.

Définition 4.7 (Anonymat probabiliste faible)**.** *Un calcul réparti $\mathcal{D}$ garantit l'anonymat probabiliste* faible*, si pour toute coalition B de nœuds malhonnêtes, pour tout nœud honnête p, pour toute*

configuration d'entrées v non-triviale *(du point de vue de* B*) et pour toute trace* D *compatible avec* v *, il existe, avec forte probabilité, une trace* D' *compatible avec une configuration d'entrées* v' *telle que (1)* D *et* D' *sont équivalentes du point de vue de* B *et (2)* v *et* v' *diffèrent pour le nœud* p*.*

Définition 4.8 (Anonymat probabiliste fort)**.** *Un calcul réparti* $\mathcal{D}$ *garantit l'anonymat probabiliste* fort*, si pour toute coalition* B *de nœuds malhonnêtes, pour tout nœud honnête* p*, pour toute configuration d'entrées* v ambiguë *(du point de vue de* B*) et pour toute trace* D *compatible avec* v *, il existe, avec forte probabilité, une trace* D' *compatible avec une configuration d'entrées* v' *telle que (1)* D *et* D' *sont équivalentes du point de vue de* B *et (2)* v *et* v' *diffèrent pour le nœud* p*.*

L'intuition derrière ces définitions est qu'il y a une forte probabilité que la coalition de nœuds malhonnêtes ne puisse pas distinguer deux exécutions différentes du calcul dans lesquelles deux nœuds honnêtes ont échangé leurs entrées. Par conséquent, l'entrée d'aucun nœud honnête n'est révélée avec certitude. Plus précisément, pour chaque nœud honnête il existe une autre configuration où son entrée est différente et qui donne la même trace du point de vue de la coalition avec la même probabilité que la véritable configuration.

En utilisant les définitions précédentes, on peut désormais définir le problème S^3 de manière formelle :

Définition 4.9 (Le problème S^3)**.** *Soit* $\mathcal{C}$ *un candidat à* S^3*. On dit qu'un protocole réparti* $(g, h, \textit{faible}) - S^3$ *calcule (respectivement* $(g, h, \textit{fort}) - S^3$ *calcule) le candidat* $\mathcal{C}$ *si le protocole est (1)* g*-extensible, (2) calcule* h*-précisément* $\mathcal{C}$ *et garantit l'anonymat probabiliste faible (respectivement fort) pour un ensemble* Π *de processus liés aux utilisateurs d'un réseau social.*

4.3.5 Réduction au problème de l'addition

Dans cette sous-section, on montre qu'un protocole capable de calculer de manière S^3 la somme composante-à-composante de vecteurs d'entiers dont toutes les composantes sont nulles sauf une qui est égale à 1 peut S^3 calculer tout candidat continu au sens de Lipschitz du moment que V est fini et que le diamètre $\Delta(N)$ est en $\Omega(N)$. Le principe de cette réduction du problème S^3 consiste à représenter le multi-ensemble des entrées des nœuds sous forme d'un vecteur d'entiers, l'union correspondant à une addition composante-à-composante. Si un protocole réparti parvient à calculer ce multi-ensemble précisément et que la fonction à calculer est suffisamment régulière, c'est-à-dire qu'une légère déviation sur le multi-ensemble de ses entrées a un impact limité sur son résultat, alors chaque nœud peut calculer localement et précisément le résultat. Dans cette sous-section on définit formellement ces notions.

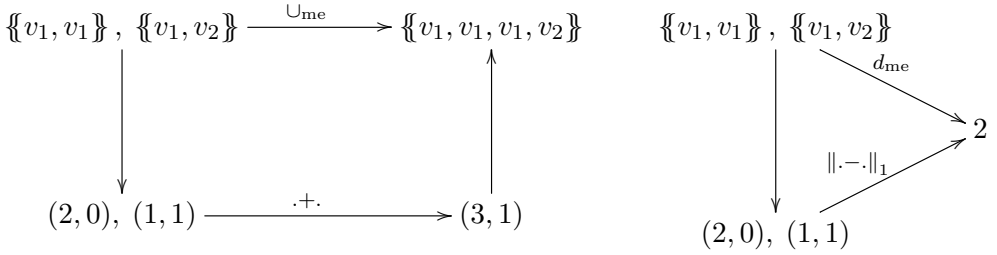
Définition 4.10 (Continuité au sens de Lipschitz)**.** *Une fonction* $f : A \to B$ *est dite continue au sens de Lipschitz pour deux mesures de distances* d_A *et* d_B *sur les ensembles* A *et* B *si le rapport de la distance entre deux éléments distincts de* A *à la distance entre leurs images respectives par* f *est borné. C'est-à-dire qu'il existe une constante* $k \in \mathbb{R}$ *telle que pour tout* x, y *dans* A *:*

$$d_B(f(x), f(y)) \leq k \cdot d_A(x, y) \ .$$

On considère maintenant un candidat à S^3 (f, V, U, d) tel qu'il a été introduit par la définition 4.1 de la sous-section précédente. Comme f est symétrique elle peut être vue comme une fonction qui, à un multi-ensemble d'éléments de V associe un élément de U. Dès lors que l'ensemble V est de

taille finie ($V = \{v_1, \ldots, v_{|V|}\}$), un multi-ensemble d'éléments de V peut être représenté comme un vecteur d'entiers naturels de taille $|V|$ dont la i-ème composante représente la multiplicité, c'est-à-dire le nombre d'occurrences, de v_i. On peut alors calculer la distance d_{me} entre deux multi-ensembles et leur union $\cup_{\mathrm{me}}$ directement sur les vecteurs d'entiers les représentant : la distance est la norme ℓ^1 de la différence des vecteurs et l'union est la somme (composante-à-composante) des vecteurs.

On considère à titre d'exemple l'ensemble $V = \{v_1, v_2\}$ et les multi-ensembles $S_1 = \{\!\{v_1, v_1\}\!\}$ et $S_2 = \{\!\{v_1, v_2\}\!\}$ d'éléments de V. Les représentations vectorielles de S_1 et S_2 sont respectivement $(2, 0)$ et $(1, 1)$. La distance $d_{\mathrm{me}}(S_1, S_2)$ est égale à $\|(2, 0) - (1, 1)\|_1 = \|(1, -1)\|_1 = |1| + |-1| = 2$ et l'union est représentée par le vecteur $(3, 1)$ qui correspond bien au multi-ensemble $S_1 \cup_{\mathrm{me}} S_2 = \{\!\{v_1, v_1, v_1, v_2\}\!\}$. Cet exemple peut être illustré par le schéma suivant :



On peut désormais définir la notion de candidat à S^3 continu au sens de Lipschitz et énoncer le théorème de réduction.

Définition 4.11 (Candidat à S^3 continu au sens de Lipschitz)**.** *Un candidat à S^3 (f, V, U, d) est dit continu au sens de Lipschitz si f est continue au sens de Lipschitz pour les distances d_{me} et d.*

Théorème 4.6 (Réduction du problème S^3)**.** *Soit $\mathcal{C} = (f, V, U, d)$ un candidat à S^3 continu au sens de Lipschitz et tel que (1) V est de taille finie, (2) le diamètre $\Delta(N)$ est en $\Omega(N)$ et (3) il existe un algorithme $\mathcal{A}$ calculant f localement à partir d'une représentation vectorielle d'un multi-ensemble de N valeurs dans V avec une complexité en $\mathcal{O}(g(N))$. S'il existe un protocole réparti $\mathcal{D}$ qui $(g, h, \textit{faible}) - S^3$ calcule la somme de vecteur d'entiers dont toutes les composantes sont nulles sauf une égale à 1, alors on peut $(g, h, \textit{faible}) - S^3$ calculer f de manière répartie.*

Démonstration. Pour prouver le théorème, on construit un protocole réparti et on prouve les propriétés d'extensibilité, de précision et de confidentialité. On considère le protocole suivant : chaque nœud p transforme son entrée v_p par la représentation vectorielle du multi-ensemble $\{\!\{v_p\}\!\}$ et on utilise le protocole réparti $\mathcal{D}$ de l'hypothèse qui permet à chaque nœud p de calculer le multi-ensemble me_p des valeurs d'entrée des nœuds sous forme de vecteurs d'entiers. Une telle représentation est possible car V est fini. Chaque nœud calcule alors localement le résultat $o_p = f(\mathrm{me}_p)$ grâce à l'algorithme $\mathcal{A}$ de l'hypothèse en utilisant directement la représentation vectorielle.

Extensibilité Le protocole de calcul réparti du multi-ensemble des valeurs d'entrée satisfait les conditions $(g, h,$ faible$)$ par hypothèse. Par conséquent sa complexité est en $\mathcal{O}(g(N))$. De même la complexité du calcul local par l'algorithme $\mathcal{A}$ est, par hypothèse, en $\mathcal{O}(g(N))$. On en conclut que la complexité du protocole proposé est en $\mathcal{O}(g(N))$.

Précision Le multi-ensemble est calculé de manière h-précise. L'erreur commise sur le résultat final est bornée par l'erreur commise sur le multi-ensemble des valeurs d'entrée multipliée

par une constante car le candidat est, par hypothèse, continu au sens de Lipschitz. D'autre part, le diamètre $\Delta_{\mathrm{me}}(N)$ pour le calcul du multi-ensemble des entrées en considérant la distance d_{me}, est égal à $2N$. Enfin, on a par hypothèse, $\Delta(N) = \Omega(N)$ pour le candidat $\mathcal{C}$. En conclusion :

$$\begin{aligned}
\frac{1}{\Delta(N)} \cdot \max_{p \text{ honnête}} d(o_p, f(v_1, \ldots, v_N)) &= \frac{1}{\Delta(N)} \cdot \max_{p \text{ honnête}} d(f(\mathrm{me}_p), f(v_1, \ldots, v_N)) \\
&\leq k \cdot \frac{1}{\Delta(N)} \cdot \max_{p \text{ honnête}} d_{\mathrm{me}}(\mathrm{me}_p, \{\!\{v_1, \ldots, v_N\}\!\})) \\
&\leq 2k \cdot \frac{N}{\Delta(N)} \cdot \frac{1}{\Delta_{\mathrm{me}}(N)} \cdot \max_{p \text{ honnête}} d_{\mathrm{me}}(\mathrm{me}_p, \{\!\{v_1, \ldots, v_N\}\!\}) \\
\frac{1}{\Delta(N)} \cdot \max_{p \text{ honnête}} d(o_p, f(v_1, \ldots, v_N)) &= \mathcal{O}(1) \cdot \mathcal{O}\left(\frac{1}{h(N)}\right) .
\end{aligned}$$

Par conséquent le protocole proposé calcule h-précisément $\mathcal{C}$.

Confidentialité On note tout d'abord que dans le cas du calcul du multi-ensemble des valeurs d'entrée la confidentialité faible est équivalente à la confidentialité forte. D'autre part, il apparaît clairement que le protocole proposé ici ne peut pas assurer la confidentialité forte : si tous les nœuds honnêtes ont la même entrée, cette information peut être déduite du multi-ensemble des valeurs d'entrée calculé. Par conséquent, le protocole ne peut être privé que pour des distributions d'entrées non-triviales. Le calcul local effectué par les nœuds n'apporte pas plus d'information aux nœuds malhonnêtes. Par conséquent la confidentialité faible du calcul du multi-ensemble des valeurs d'entrée implique la confidentialité faible du protocole proposé.

□

Inspiré par le protocole de sondage binaire DPol proposé dans la section précédente et motivé par le théorème de réduction, on s'intéresse dans la suite au problème $(\sqrt{}, \sqrt{}, \text{faible}) - S^3$ permettant de calculer la somme de vecteurs d'entiers dont toutes les composantes sont nulles sauf une qui est égale à 1.

4.4 Agg-S^3 : S^3 calcul de fonctions d'agrégation

Cette section présente Agg-S^3, un protocole de calcul réparti qui résout le problème $(\sqrt{}, \sqrt{}, \text{faible}) - S^3$ pour une classe de fonctions d'agrégation en présence de $|B| \leq \sqrt{N}/\log^2 N$ nœuds malhonnêtes. Agg-S^3 généralise les techniques utilisées dans DPol de manière à calculer un plus grand nombre de fonctions et apporte des modifications clés à la structure permettant de satisfaire les propriétés de sécurité du problème S^3 tout en restant $\sqrt{}$-extensible.

4.4.1 Hypothèses

Cette sous-section décrit la classe de fonctions considérée dans la conception de Agg-S^3 et donne, pour chaque propriété, une intuition de la raison pour laquelle cette dernière facilite ou rend possible le calcul S^3.

On considère les candidats à S^3 pour lesquels la fonction f dérive d'une loi interne $\oplus$ associative sur U, c'est-à-dire que $\oplus$ est un opérateur binaire de $U \times U \to U$ tel que $(v_1 \oplus v_2) \oplus v_3 = v_1 \oplus (v_2 \oplus v_3)$ pour tout v_1, v_2 et v_3 dans U et que f est une fonction d'agrégation que l'on écrit :

$f(v_1, \ldots, v_N) = v_1 \oplus \cdots \oplus v_N$. La fonction f étant par définition symétrique, on en déduit que l'opérateur $\oplus$ est commutatif. Ces propriétés induisent une structure de monoïde commutatif sur $(U, \oplus)$ (on note 0_U son élément neutre) et V est nécessairement un sous-ensemble de U. La classe des fonctions d'agrégation est particulièrement intéressante dans le contexte des systèmes répartis car leur calcul est facilement parallélisable : la propriété d'associativité permet de découper le calcul en sous-calculs et la propriété de commutativité permet d'effectuer les sous-calculs sur n'importe quelle partition du multi-ensemble des valeurs d'entrée. Plus formellement, pour toute partition $S_1, \ldots, S_m$ de l'ensemble $\{v_1, \ldots, v_N\}$, on a $f(v_1, \ldots, v_N) = \bigoplus_{i=1}^{m} \left(\bigoplus_{v \in S_i} v\right)$.

On suppose d'autre part que l'opérateur $\oplus$ est *compatible* avec la distance d dans le sens où pour tout v_1, v_2, v'_1 et v'_2 dans U on a :

$$d(v_1 \oplus v_2, v'_1 \oplus v'_2) \leq d(v_1, v'_1) + d(v_2, v'_2) \ . \tag{4.2}$$

On remarquera ici que le candidat à S^3 $((v_1, \ldots, v_N) \mapsto v_1 + \cdots + v_N, \{-1, 1\}, \mathbb{Z}, (z_1, z_2) \mapsto |z_1 - z_2|)$ est une agrégation et l'opérateur $+$ dont il dérive est compatible avec la distance utilisée. En revanche, la fonction calculant la somme des produits deux-à-deux des entrées des nœuds $f(v_1, \ldots, v_N) = v_1 \cdot v_2 + \cdots + v_1 \cdot v_N + \cdots + v_{N-1} \cdot v_N$ ne peut être exprimée comme une fonction d'agrégation. La propriété de compatibilité facilite les vérifications car elle donne une borne sur la distance entre deux sous-résultats.

L'ensemble des valeurs d'entrée possibles V est supposé indépendant de N et de taille finie et la taille de l'ensemble d'arrivée U est supposée être bornée par un polynôme en N. La première hypothèse assure que le diamètre δ_V de V, $\delta_V = \max_{v_1, v_2 \in V} d(v_1, v_2)$, est fini et constant (indépendant de N). La seconde hypothèse implique que la taille d'un message contenant une valeur d'entrée ou un résultat intermédiaire est logarithmique en N.

On supposera également que la diamètre de l'espace image de f est au moins aussi grand que N, c'est-à-dire que $\Delta(N) = \Omega(N)$. Cette propriété assure que, si l'erreur commise par un protocole de calcul réparti sur le résultat est bornée par une constante, alors l'erreur relative tend vers 0 et donc le protocole est h-précis pour une certaine fonction h qui tend vers l'infini quand N tend vers l'infini. Cette hypothèse exclut, par exemple, les fonctions d'agrégation binaires, c'est-à-dire à valeurs dans $\{0, 1\}$.

Enfin, on supposera que l'ensemble V est clos pour l'opposé vis-à-vis de l'opérateur $\oplus$, c'est-à-dire que, pour chaque élément v de V, il existe un élément noté $\ominus$ dans V tel que $v \oplus (\ominus v) = 0_U$. Cette hypothèse permet de construire un mécanisme simple de partage de secret par homomorphisme car un nœud peut « annuler » un bulletin émis en envoyant un bulletin contenant son opposé.

4.4.2 Le protocole Agg-S^3

Le principal défi du calcul S^3 est la dualité entre l'extensibilité et la précision d'une part et la confidentialité d'autre part. Intuitivement, pour garantir la confidentialité sans cryptographie, un nœud doit partager sa valeur d'entrée en plusieurs bulletins, ce qui augmente le nombre de messages circulant dans le système et donc réduit l'extensibilité du protocole et augmente l'impact potentiel des nœuds malhonnêtes réduisant ainsi la précision. D'autre part, l'utilisation de vérifications, nécessitant des échanges de messages, pour garantir la précision en exploitant l'aspect social des utilisateurs, fait apparaître la dualité entre extensibilité et précision. On décrit dans la suite la structure utilisée par Agg-S^3 ainsi que les compromis faits par ce dernier pour garantir les trois propriétés fondamentales du problème S^3 puis on détaille le protocole Agg-S^3. L'algorithme 2

donne une version en pseudo-code de Agg-S^3 incluant le protocole d'agrégation et les procédures de vérification.

Structure Comme pour DPol, la propriété de $\sqrt{}$-extensibilité de Agg-S^3 est obtenue à l'aide d'un réseau couvrant qui partitionne les nœuds du système en groupes de taille $\sqrt{N}$ disposés sur un anneau. Un nœud n'échange des messages qu'avec les autres nœuds de son groupe et *quelques* nœuds des groupes *proches*. Plus concrètement, on définit deux paramètres κ et l pour quantifier ces notions : un nœud communique avec l autres nœuds, appelés intermédiaires, dans chacun des κ groupes succédant au sien sur l'anneau. La figure 4.8 représente le réseau couvrant utilisé par Agg-S^3. On utilise la même terminologie que DPol pour les relations entre les nœuds : assesseurs pour les membres d'un même groupe et intermédiaires/clients pour les nœuds dans des groupes différents. On supposera qu'un nœud a exactement $\kappa \cdot l$ clients.

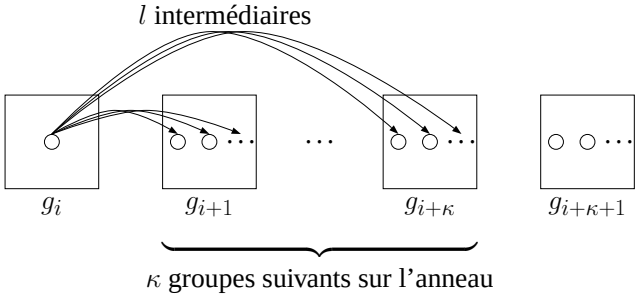


FIGURE 4.8 : Réseau couvrant utilisé par Agg-S^3, centré sur un utilisateur du groupe g_i.

Pour des raisons d'extensibilité, les paramètres κ et l doivent être petits car ils sont directement liés au nombre de messages envoyés. De plus, si κ et l sont grands, un nœud malhonnête reçoit plus de messages et donc il a plus d'opportunités de biaiser le résultat du calcul. Pour garantir la confidentialité en revanche, il faut qu'il existe un mécanisme transformant une trace de message en une autre trace telle que deux nœuds honnêtes aient échangé leurs entrées tout en assurant que les messages reçus par les nœuds malhonnêtes restent les mêmes. Ainsi, la confidentialité de l'entrée de p est assurée. Il apparaîtra clairement dans la preuve de confidentialité que de grandes valeurs de κ et l sont nécessaires pour garantir l'existence d'une telle transformation avec une forte probabilité. Dans Agg-S^3, le compromis entre extensibilité, précision et confidentialité est atteint pour des valeurs de κ et l en $\Theta(\log N)$: plus précisément on fixe $\kappa = 3/2 \cdot \lfloor \ln N \rfloor$ et $l = 5 \cdot \lfloor \ln N \rfloor$. On suppose que κ et l sont impairs. Les choix de ces paramètres seront justifiés dans la suite.

Agrégation Tout comme DPol, Agg-S^3 s'exécute en trois phases : une phase de partage de la valeur d'entrée et d'envoi des bulletins, une phase de comptage local dans chaque groupe et une phase de dissémination des résultats intermédiaires.

Durant la première phase, un nœud partage son vote en $\kappa \cdot l$ bulletins et en envoie un à chacun de ses intermédiaires (un nœud ayant l intermédiaires dans chacun des κ groupes suivant le sien sur l'anneau, il a donc $\kappa \cdot l$ intermédiaires au total). Le partage de l'entrée d'un nœud p consiste à construire un ensemble de $(\kappa \cdot l - 1)/2$ valeurs choisies aléatoirement et uniformément dans V auxquelles on ajoute leurs opposés respectifs et enfin l'entrée v_p du nœud, soit un total de $2(\kappa \cdot l - 1)/2 + 1 = \kappa \cdot l$ éléments dont la somme est égale à v_p.

Algorithme 2 Protocole Agg-S^3

Entrées : une valeur d'entrée $v \in V$
Variables : agrégat individuel $u'' = 0_U$
agrégat local $u' = 0_U$

procédure vote(v) **:**
1: **pour** $i = 1$ **à** $(l \cdot \kappa - 1)/2$ **faire**
2: $b_i = \mathrm{alea}(V)$ # choix d'une valeur aléatoire dans V (uniformément)
3: $b_{i+(\kappa \cdot l-1)/2} = \ominus s_i$ # ajout de l'opposé
4: **fin pour**
5: $b_{l \cdot \kappa} = v$ # ajout de la valeur d'entrée
6: $\sigma = \mathrm{rand}(\mathcal{S}_{l \cdot \kappa})$ # distribution des bulletins suivant une permutation aléatoire
7: **pour** $i_{groupe} = 1$ **à** κ **faire**
8: **pour** $i_{intermédiaire} = 1$ **à** l **faire**
9: envoyer[BULLETIN, $b_{\sigma(i_{groupe} \cdot l + i_{intermédiaire})}$]($p_{i_{groupe}, i_{intermédiaire}}$)
10: **fin pour**
11: **fin pour**

lors de l'événement ⟨ réception | [BULLETIN, b] ⟩ **faire**
12: vérifier que l'expéditeur est bien un client
13: vérifier que la valeur reçue est valide # $b \in V$
14: $u'' = u'' \oplus b$

procédure comptage_local(u'') **:**
15: **pour tout** $assesseur$ **faire**
16: envoyer[AGRÉGATLOCAL,u'']($assesseur$)
17: **fin pour**

lors de l'événement ⟨ réception | [AGRÉGATLOCAL, u] ⟩ **faire**
18: vérifier que l'expéditeur est bien un assesseur
19: vérifier que l'agrégat reçu peut être l'agrégation de $\kappa \cdot l$ valeurs d'entrée
20: # $d(u, w_1 \oplus \cdots \oplus w_{\kappa \cdot l}) \leq \kappa \cdot l \cdot \delta_V$ où les $w_1 \oplus \cdots \oplus w_{\kappa \cdot l}$ sont des valeurs aléatoires de V
21: $u' = u' \oplus u$

Pendant la phase de comptage intermédiaire, chaque nœud agrège les bulletins qu'il a reçus de ses clients, c'est-à-dire qu'il calcule leur somme (pour l'opérateur $\oplus$). Il transmet l'agrégat individuel ainsi calculé à ses assesseurs qui font de même. Chaque nœud somme alors les agrégats individuels qu'il reçoit de ses assesseurs auquel il ajoute le sien, ce qui donne un agrégat local.

Enfin, la phase de dissémination consiste à faire circuler un *jeton*, c'est-à-dire un message transmis de nœud en nœud, le long de l'anneau auquel vont être ajoutés les agrégats locaux calculés dans les groupes. Un groupe, celui d'indice zéro par exemple, est désigné pour initier la diffusion du jeton. Chaque membre de ce groupe crée un jeton, initialisé avec l'agrégat local calculé dans le groupe et le transmet à tous ses l intermédiaires dans le groupe suivant, et seulement dans celui-là, sur l'anneau. Lorsqu'un nœud reçoit le jeton durant son premier tour sur l'anneau, il y ajoute l'agrégat local calculé dans son groupe et le transmet à ses intermédiaires dans le groupe suivant. Lorsqu'un nœud reçoit le jeton lors de son second tour sur l'anneau, il prend la valeur du jeton comme résultat. Les nœuds du groupe ayant initié la diffusion du jeton incrémentent le compteur de tours sur le jeton avant de le transmettre. Quand ce compteur dépasse 2, le jeton n'est pas retransmis. L'utilisation d'un seul jeton (plutôt qu'un jeton par groupe comme cela est fait dans DPol) permet d'assurer l'extensibilité des vérifications, comme expliqué dans le paragraphe suivant.

Vérifications Les vérifications sont exécutées parallèlement au protocole et conduisent chaque nœud à rapporter, via l'infrastructure du réseau social, l'ensemble des nœuds qu'il détecte comme malhonnêtes. On associe à chaque phase une procédure de vérification. Dans la phase de vote, les intermédiaires se contentent de vérifier que les bulletins qu'ils reçoivent sont effectivement des éléments de V. On notera ici l'importance du fait que la méthode de partage de secrets par homomorphisme utilisée génère des bulletins à valeurs dans V. Dans la phase de comptage local, un nœud veut vérifier que les agrégats partiels envoyés par ses assesseurs résultent de l'agrégation de $\kappa \cdot l$ bulletins à valeurs dans V. Cette vérification s'appuie sur la propriété suivante qui découle directement (par récurrence sur k) de la compatibilité entre la distance d et l'opérateur $\oplus$ (voir équation 4.2)) : pour tout entier k et pour tout $w_1, \ldots, w_k, w'_1, \ldots, w'_k$ dans V,

$$d(w_1 \oplus \cdots \oplus w_k, w'_1 \oplus \cdots \oplus w'_k) \leq d(w_1, w'_1) + \cdots + d(w_k, w'_k) \leq k \cdot \delta_V \ . \tag{4.3}$$

Le principe de la vérification consiste à évaluer la distance entre l'agrégat individuel reçu par le nœud et une combinaison quelconque (choisie aléatoirement par exemple) de valeurs d'entrée. On notera que cela ne garantit en rien que l'agrégat individuel est effectivement la somme de $\kappa \cdot l$ valeurs d'entrée, cependant cela permet de borner la distance entre la somme des bulletins reçus par l'intermédiaire et l'agrégat individuel qu'il diffuse à ses assesseurs. Enfin, lors de la phase de dissémination, la vérification a pour but d'assurer que le jeton n'est pas corrompu. Comme dans DPol, chaque nœud s'assure que les jetons reçus de ses clients ont la même valeur et que ses assesseurs ont reçu cette même valeur.

4.4.3 Preuve de correction

Théorème 4.7 (Calcul S^3 de fonctions d'agrégation par Agg-S^3)**.** *Le protocole Agg-S^3 $(\sqrt{}, \sqrt{}, faible) - S^3$-calcule les fonctions d'agrégation vérifiant les propriétés énoncées dans la sous-section 4.4.1 en présence de $|B| \leq \sqrt{N}/\log^2 N$ nœuds malhonnêtes liés à un réseau social.*

Cette sous-section prouve ce théorème principal en montrant que Agg-S^3 satisfait les trois propriétés fondamentales du problème S^3 sous ces hypothèses.

Théorème 4.8 (Extensibilité de Agg-S^3)**.** *Le protocole Agg-S^3 est $\sqrt{}$-extensible.*

Démonstration. Un nœud maintient la liste des identifiants (de taille $\log N$) de ses $\sqrt{N}$ assesseurs, de ses $\kappa \cdot l = \mathcal{O}(\log^2 N)$ intermédiaires et de ses $\kappa \cdot l$ clients Un nœud doit également maintenir les agrégats individuels (de taille $\log N$) de ses $\sqrt{N}$ assesseurs afin de les comparer pour les vérifications. La complexité en mémoire est donc en $\mathcal{O}(\sqrt{N} \cdot \log N)$. De même la complexité en termes de calcul est en $\mathcal{O}(\sqrt{N} \cdot \log N)$. La complexité en termes de messages quant à elle est en $\mathcal{O}(\sqrt{N})$ (pour une taille de message de $\log N$) du fait des échanges suivants : un nœud envoie $\kappa \cdot l$ bulletins, son agrégat individuel à ses $\sqrt{N}$ assesseurs, deux fois un jeton à ses $\kappa \cdot l$ intermédiaires (pour la diffusion) et à ses assesseurs (pour la vérification). Puisque la relation d'assesseur est symétrique et qu'un nœud a autant de clients que d'intermédiaires, il reçoit autant de messages qu'il en envoie. □

Théorème 4.9 (Précision de Agg-S^3)**.** *Le protocole Agg-S^3 est $\sqrt{}$-précis en présence de $|B| \leq \sqrt{N}/\log^2 N$ nœuds malhonnêtes liés à un réseau social.*

Démonstration. Un nœud malhonnête peut biaiser le résultat du calcul en envoyant un ensemble invalide de bulletins, c'est-à-dire dont la somme n'appartient pas à V, ou en envoyant à ses assesseurs un agrégat individuel différent de la somme des bulletins qu'il a reçus, ou encore en changeant la valeur du jeton pendant la phase de diffusion. Cependant, un nœud malhonnête se limite aux déviations qui ne sont pas détectées à coup sûr.

Vote : Pour ne pas être être détecté, un nœud malhonnête ne doit envoyer que des bulletins à valeurs dans V et uniquement à ses intermédiaires. Par conséquent, un nœud malhonnête ne peut envoyer que $\kappa \cdot l$ bulletins, et donc, en utilisant l'inégalité (4.3), on peut borner son impact maximum. En effet, si un nœud envoie $\kappa \cdot l$ bulletins de valeurs $w_1, \ldots, w_{\kappa \cdot l}$ alors l'écart minimum entre la somme des bulletins qu'il envoie et un vote valide est au plus :

$$d(w_1 \oplus \cdots \oplus w_{\kappa \cdot l}, w_1 \oplus \cdots \oplus w_{(\kappa \cdot l - 1)/2} \ominus w_1 \cdots \ominus w_{(\kappa \cdot l - 1)/2)} \oplus w_{\kappa \cdot l}) \leq \frac{(\kappa \cdot l - 1)}{2} \cdot \delta_V \ .$$

Intuitivement, cette borne vient du fait que seuls les $(\kappa \cdot l - 1)/2$ bulletins contenant les opposés sont « imposés » dans la méthode de partage de secret par homomorphisme utilisée.

Comptage local : On considère un nœud malhonnête qui reçoit les bulletins $w_1, \ldots, w_{\kappa \cdot l}$ et envoie un agrégat local erroné u à ses assesseurs. Ces derniers vérifient que la distance entre u et un certain $w' = w'_1 \oplus \ldots w'_{\kappa \cdot l}$ choisi aléatoirement est inférieure à $\kappa \cdot l \cdot \delta_V$. En utilisant à nouveau l'inégalité (4.3) et l'inégalité triangulaire pour la distance d, on obtient une borne sur l'impact d'un nœud malhonnête :

$$\begin{aligned} d(w_1 \oplus \cdots \oplus w_{\kappa \cdot l}, u) &\leq d(w_1 \oplus \cdots \oplus w_{\kappa \cdot l}, w'_1 \oplus \cdots \oplus w'_{\kappa \cdot l}) + d(w'_1 \oplus \cdots \oplus w'_{\kappa \cdot l}, u) \\ &\leq \kappa \cdot l \cdot \delta_V + \kappa \cdot l \cdot \delta_V \\ d(w_1 \oplus \cdots \oplus w_{\kappa \cdot l}, u) &\leq 2\kappa \cdot l \cdot \delta_V \end{aligned}$$

Diffusion : On a montré dans l'analyse de DPol (lemme 4.4, page 58) que pour modifier la valeur d'un jeton (un scrutin local dans le cas de DPol) sans être détectés, les nœuds malhonnêtes doivent représenter plus de la moitié des nœuds dans l'union de deux groupes consécutifs. Cela nécessite un nombre total de nœud malhonnêtes au moins égal à la taille d'un groupe, c'est-à-dire $\sqrt{N}$. L'impact des nœuds malhonnêtes lors de la phase de diffusion est donc réduit à zéro sous l'hypothèse $|B| \leq \sqrt{N}/\log^2 N$.

On en conclut que l'impact maximal d'un nœud malhonnête est borné par $5/2 \cdot \kappa \cdot l \cdot \delta_V$. L'impact total d'une coalition de $|B| \leq \sqrt{N}/\log^2 N$ nœuds malhonnêtes est donc lui-même borné comme suit :

$$\max_{p \text{ honnête}} d(o_p, f(v_1, \ldots, v_N)) \leq \frac{5}{2} \kappa \cdot l \cdot \delta_V \cdot \frac{\sqrt{N}}{\log^2 N}$$

avec $\kappa \cdot l = \mathcal{O}(\log^2 N)$ et $\delta_V = \mathcal{O}(1)$. En intégrant le fait que $\Delta(N) = \Omega(N)$ on obtient alors :

$$\frac{1}{\Delta(N)} \cdot \max_{p \text{ honnête}} d(o_p, f(v_1, \ldots, v_N)) = \mathcal{O}\left(\frac{1}{\sqrt{N}}\right) \ ,$$

ce qui conclut le théorème. □

Théorème 4.10 (Anonymat probabiliste faible dans Agg-S^3)**.** *Le protocole Agg-S^3 garantit l'anonymat probabiliste faible en présence de $|B| \leq \sqrt{N}/\log^2 N$ nœuds malhonnêtes.*

Démonstration. Le principe de cette preuve consiste à montrer de manière constructive que, pour chaque nœud honnête, il existe une transformation permettant de changer sa valeur d'entrée sans changer le contenu des messages reçus par la coalition de nœuds malhonnêtes. Le but étant ici de prouver la propriété d'anonymat probabiliste *faible*, on considère une distribution non-triviale d'entrées, c'est-à-dire dans laquelle les nœuds honnêtes n'ont pas tous la même entrée. On décrit dans un premier temps le principe et la structure de la preuve et on identifie les propriétés clés que l'on démontre dans les lemmes 4.5 et 4.6.

On considère une exécution D de Agg-S^3 dans laquelle un nœud honnête p a une entrée v_p. On veut montrer que l'anonymat probabiliste est garanti pour le nœud p en présence d'une coalition de $|B| < \sqrt{N}/\log^2 N$ nœuds malhonnêtes. Pour ce faire, on cherche à modifier la trace de messages de sorte que p ait une entrée différente et que les nœuds malhonnêtes reçoivent les même messages que dans la trace originale. L'hypothèse selon laquelle la distribution d'entrée est non-triviale assure que, pour chaque nœud honnête, il existe un autre nœud honnête ayant une entrée différente. Soit p' un nœud honnête dont l'entrée $v_{p'}$ dans la trace originale est différente de v_p. On cherche alors à modifier la trace originale de telle sorte que les entrées de tous les nœuds restent identiques sauf celles de p et p' qui sont interverties. On procède en plusieurs étapes.

Les nœuds malhonnêtes connaissent les bulletins qu'ils envoient, les bulletins qu'ils reçoivent, les agrégats individuels de tous les nœuds des groupes contenant au moins un nœud malhonnête et les agrégats locaux de tous les groupes. On dit qu'un groupe est non-compromis lorsqu'il ne contient pas de nœuds malhonnêtes. Par conséquent, la seule information sur un tel groupe connue des nœuds malhonnêtes est son agrégat local. On montre tout d'abord que s'il existe un groupe non-compromis g dans lequel p et p' ont des intermédiaires, alors en modifiant les messages envoyés par p et p' dans ce groupe on peut intervertir leurs entrées avec une forte probabilité sans changer l'agrégat local du groupe. Cette trace est donc équivalente à la trace originale du point de vue de la coalition de nœuds malhonnêtes. On montre ensuite que pour deux nœuds dans des groupes adjacents sur l'anneau, il existe avec une forte probabilité un groupe non-compromis dans lequel ils ont tous les deux des intermédiaires. À ce point, on aura donc montré que l'anonymat probabiliste de deux nœuds honnêtes ayant des entrées différentes et situés dans des groupes adjacents est garanti. On étendra ensuite le résultat à deux nœuds situés dans des groupes quelconques en construisant une *chaîne* d'échanges d'entrées.

On commence par prouver les lemmes suivants. On notera que ces derniers illustrent la nécessité d'avoir les paramètres κ et l grands et justifient le choix de leurs valeurs.

Lemme 4.5. *La probabilité que toutes les suites de κ-1 groupes consécutifs sur l'anneau contiennent au moins un groupe non-compromis est au moins* $1 - \sqrt{N}\left(\frac{|B|}{\sqrt{N}}\right)^{\kappa-1}$.

Démonstration. Le nombre total de nœuds malhonnêtes étant fixé, le nombre de groupes compromis, et donc cette probabilité, est minimisé quand chaque groupe contient au plus un nœud malhonnête. Dans ce cas, il y a $|B|$ groupes compromis. On considère une suite donnée de $\kappa - 1$ groupes consécutifs et on calcule la probabilité qu'ils soient tous compromis par dénombrement en supposant que les placements des groupes compromis sur l'anneau sont équiprobables. Le nombre de situations où ces groupes sont compromis est $\binom{\sqrt{N}-\kappa+1}{|B|-\kappa+1}$ pour un nombre total de $\binom{\sqrt{N}}{|B|}$ situations possibles pour le placement des groupes compromis sur l'anneau. Par conséquent, la probabilité que les $\kappa-1$ groupes consécutifs sur l'anneau soient compromis est le rapport des deux coefficients binomiaux $\binom{\sqrt{N}-\kappa+1}{|B|-\kappa+1}/\binom{\sqrt{N}}{|B|} = |B|/\sqrt{N} \times \cdots \times (|B|-\kappa+2)/(\sqrt{N}-\kappa+2)$ qui est borné par

$(|B|/\sqrt{N})^{\kappa-1}$. Sur un anneau formé de $\sqrt{N}$ groupes, il existe $\sqrt{N}$ suites de $\kappa-1$ groupes successifs. On utilise l'inégalité de Boole pour borner la probabilité qu'au moins une suite de $\kappa-1$ groupes compromis apparaisse sur l'anneau ce qui, en passant au complémentaire, prouve le lemme. □

Lemme 4.6. *Pour un $v \in V$ donné, la probabilité qu'un nœud honnête envoie au moins un bulletin de valeur v à un intermédiaire dans un groupe donné, en supposant que ce nœud a des intermédiaires dans ce groupe, est au moins de $1 - 1/N^{5/2}$.*

Démonstration. Les l bulletins envoyés dans un groupe donné par un nœud honnête sont choisis aléatoirement et uniformément dans un ensemble de $\kappa \cdot l$ valeurs dont $(\kappa \cdot l - 1)/2$ sont choisies aléatoirement, $(\kappa \cdot l - 1)/2$ sont les opposés des valeurs choisies aléatoirement, et une est l'entrée du nœud. Parmi les l valeurs envoyées dans le groupe, au moins $(l-1)/2$ sont donc indépendantes et choisies aléatoirement et uniformément dans V (le pire cas étant quand le nœud envoie dans ce groupe : son entrée, $(l-1)/2$ valeurs aléatoires et leurs opposés). Par conséquent, la probabilité que v ne soit pas l'une d'entre elles est au plus $(1 - 1/|V|)^{(l-1)/2}$. Comme $(l-1)/2 = 3 \cdot |V| \cdot \lfloor \log N \rfloor$, alors cette probabilité est inférieure à $1/N^{5/2}$, ce qui prouve le lemme. □

Considérons maintenant deux nœuds honnêtes p et p' dans des groupes consécutifs tels que $v_p \neq v_{p'}$. Chacun d'entre eux a l intermédiaires dans les κ groupes suivant le sien sur l'anneau. Par conséquent, il y a $\kappa - 1$ groupes dans lesquels tous les deux ont des intermédiaires. On suppose qu'au moins un de ces groupes soit non-compromis (le lemme 4.5 donne une borne inférieure sur la probabilité de cet événement). On appelle g ce groupe. On suppose également que parmi les bulletins envoyés par p (respectivement p') à ses intermédiaires dans g, au moins un contient la valeur v_p (respectivement $v_{p'}$) (le lemme 4.6 donne une borne inférieure sur la probabilité de cet événement). On considère la trace où un bulletin de valeur v_p envoyé par p à un de ses intermédiaires dans g est remplacé par un bulletin de valeur $v_{p'}$ et où un bulletin de valeur $v_{p'}$ envoyé par p' à un de ses intermédiaires dans g est remplacé par un bulletin de valeur v_p. Cette trace a la même probabilité que la trace originale et est compatible avec la configuration où les entrées des nœuds restent les mêmes sauf celles de p et p' qui sont interverties. De plus cette trace est équivalente à la trace originale du point de vue des nœuds malhonnêtes car g est non-compromis et son agrégat local est inchangé. Cette transformation est illustrée sur la figure 4.9.

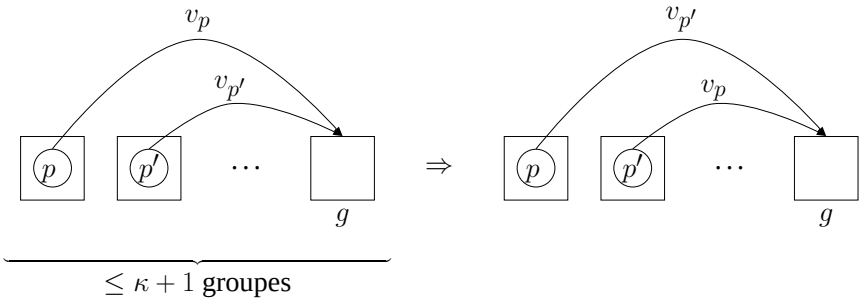
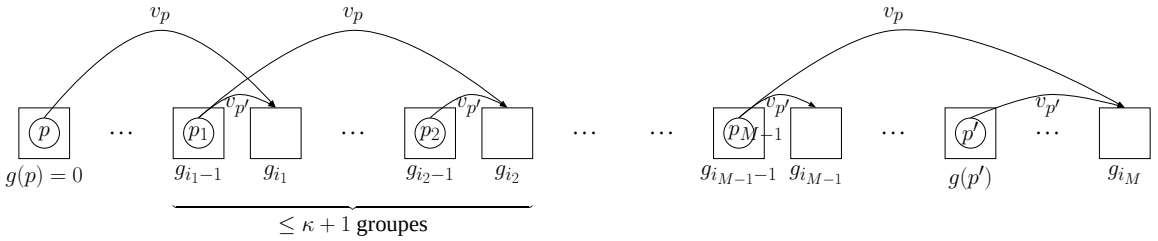


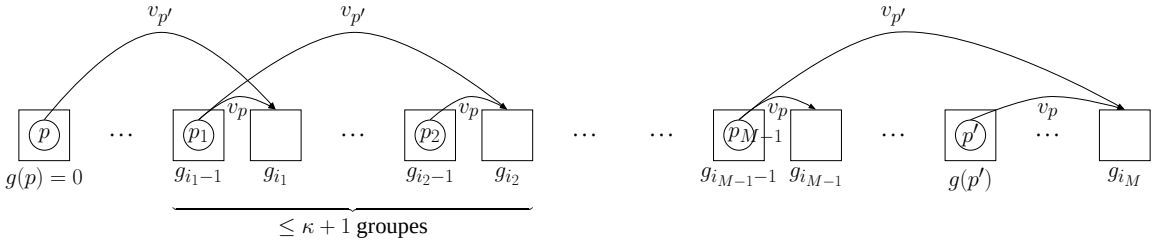
FIGURE 4.9 : Transformation d'une trace par échange d'entrées dans un groupe non-compromis.

On étend maintenant ce résultat au cas où p et p' sont dans deux groupes quelconques en construisant une chaîne de transformations telles que celle illustrée sur la figure 4.9. On note $g(\cdot)$ la fonction

qui, à un nœud associe l'indice du groupe auquel il appartient. On suppose sans perte de généralité que $g(p) = 0$. Soit $i_1, \ldots, i_M$ une suite d'indices de groupes telle que : (1) le groupe g_{i_m} est non-compromis pour tout $m \in \{1, \ldots, M\}$, (2) $0 < i_1 \leq \kappa$, (3) $0 < i_{m+1} - i_m < \kappa$ pour tout $m \in \{1, \ldots, M-1\}$ et (4) $0 < i_M - g(p') < \kappa$. Une telle suite d'indices de groupes existe avec une forte probabilité d'après le lemme 4.5. Pour tout $1 \leq m < M$, on désigne par p_m un nœud honnête quelconque du groupe $g_{i_m - 1}$. Un tel nœud existe car le nombre de nœuds malhonnêtes est inférieur à la taille d'un groupe. Enfin, on fixe $p_0 = p$ et $p_M = p'$. Comme tous les nœuds ont des intermédiaires dans les κ groupes succédant au leur sur l'anneau, on obtient que pour tout $1 \leq m \leq M$, p_{m-1} et p_m ont des intermédiaires communs dans g_{i_m} (qui est non-compromis), comme illustré sur la figure 4.10.



(a) Configuration initiale



(b) Configuration après une transposition d'entrées

FIGURE 4.10 : Illustration de la preuve de l'anonymat probabiliste faible (théorème 4.10) : en échangeant deux-à-deux des bulletins envoyés dans un groupe non-compromis ((a) $\rightarrow$ (b)) on obtient une trace équivalente du point de vue des nœuds malhonnêtes mais dans laquelle les entrées de deux nœuds ont été échangées.

D'après le lemme 4.6 et en utilisant l'inégalité de Boole, on obtient que la probabilité que, pour tout $1 \leq m \leq M$, p_{m-1} envoie un bulletin de valeur v_p à un intermédiaire dans g_m et que p_m envoie un bulletin de valeur v_p à un intermédiaire dans g_m, est au moins $1 - 2M/N^{5/2}$. Comme M est borné par le nombre de groupes, c'est-à-dire $\sqrt{N}$, cette probabilité est supérieure à $1 - 2/N^2$. On a donc montré qu'il existe avec une forte probabilité une situation telle que celle illustrée sur la figure 4.10a. On transforme alors la trace d'exécution initiale en remplaçant le bulletin de valeur v_p envoyé par p dans g_{i_1} par $v_{p'}$, en remplaçant le bulletin de valeur $v_{p'}$ envoyé par p' dans g_{i_M} par v_p et en intervertissant, pour tout $m \in \{1, \ldots, M-1\}$, les bulletins de valeurs v_p et $v_{p'}$ envoyés par p_m dans g_{i_m} et g_{i_m+1} respectivement. La trace ainsi obtenue conserve les agrégats locaux des groupes d'indices $i_1, \ldots, i_M$ qui sont non-compromis. Elle est donc équivalente à la trace originale du point de vue des nœuds malhonnêtes. D'autre part elle est compatible avec la configuration d'entrée modifiée où toutes les entrées restent les mêmes sauf celles de p et p' qui sont interverties.

En conclusion, on a montré que si le lemme 4.5 est vérifié, l'anonymat probabiliste de p et p' est garanti. En utilisant la règle de Bayes, on obtient que l'anonymat d'un nœud donné est garanti avec une probabilité supérieure à $1 - 3/N^2$. En utilisant l'inégalité de Boole sur l'ensemble des $N - B \leq N$ nœuds honnêtes, on obtient que Agg-S^3 assure l'anonymat de tous les nœuds honnêtes avec une probabilité supérieure à $1 - 1/N$ ce qui prouve la propriété d'anonymat probabiliste au sens du problème S^3, et donc le théorème. □

4.4.4 Généralisation

On a montré, lors de la présentation du problème S^3, que si un protocole réparti $(g, h, \text{faible}) - S^3$ calcule le multi-ensemble des valeurs d'entrée des nœuds sous forme vectorielle, alors il peut $(g, h, \text{faible}) - S^3$ calculer les candidats à S^3 vérifiant des conditions de régularité (voir théorème 4.6, page 71). On montre dans cette sous-section que Agg-S^3 vérifie les conditions du théorème de réduction.

Soit V l'ensemble des valeurs d'entrée du candidat à S^3 considéré. On utilise la représentation vectorielle des multi-ensembles de valeurs de V présentée dans la section précédente. Les valeurs d'entrée des nœuds sont donc des vecteurs d'entiers de taille $|V|$ ne contenant que des 0 sauf pour une composante qui est égale à 1. On définit l'ensemble V' des vecteurs d'entiers relatifs de taille $|V|$ dont toutes les composantes sont égales à 0 sauf une qui vaut soit -1 soit 1. On définit alors l'ensemble U' des vecteurs d'entiers relatifs de $|V|$ dont les composantes sont inférieures à $N \cdot \kappa \cdot l$ en valeur absolue ($U' = \{-N \cdot \kappa \cdot l, \ldots, N \cdot \kappa \cdot l\}^{|V|}$), que l'on munit de la loi interne $\oplus$ définie par :

$$(x_1, \ldots, x_{|V|}) \oplus (y_1, \ldots, y_{|V|}) = \left((x_1 + y_1)_{/[-N\cdot\kappa\cdot l, N\cdot\kappa\cdot l]}, \ldots, (x_{|V|} + y_{|V|})_{/[-N\cdot\kappa\cdot l, N\cdot\kappa\cdot l]}\right) \ ,$$

où $(x)_{/[-N\cdot\kappa\cdot l, N\cdot\kappa\cdot l]} = \max\left(\min\left(x, -N \cdot \kappa \cdot l\right), N \cdot \kappa, l\right)$. En bref, $\oplus$ représente l'addition composante-à-composante de deux vecteurs d'entiers relatifs de taille $|V|$ où la valeur des composantes est tronquée si elle dépasse les bornes de l'intervalle $[-N \cdot \kappa \cdot l, N \cdot \kappa \cdot l]$. On notera que cette troncature n'est jamais appliquée dans Agg-S^3 car chaque nœud n'émet que $\kappa \cdot l$ bulletins, qui sont des vecteurs dont toutes les composantes sont inférieures ou égale à 1 en valeur absolue. À la fin de l'exécution de Agg-S^3, les composantes négatives du résultat sont mise à zéro et les composantes positives sont tronquées à N pour obtenir une représentation valide d'un multi-ensemble et qui est de taille en $\mathcal{O}(N)$. Cette opération ne diminue pas la précision du résultat pour la distance $\|. - .\|_1$ car les composantes du résultat réel sont positives et leur somme est égale à N.

On peut maintenant prouver que $(V', U', \oplus, \|. - .\|_1)$ est un candidat à S^3 vérifiant les conditions de calcul par Agg-S^3. Il est facile de voir que V' est un ensemble de taille finie, indépendant de N et fermé par l'opposé, l'opposé d'un vecteur de V étant le vecteur dont toutes les composantes sont remplacées par leur opposé (au sens des entiers). L'ensemble U' est de taille $(2N \cdot \kappa \cdot l + 1)^{|V|}$, qui est donc bornée par un polynôme en N car κ et l sont en $\mathcal{O}(\log N)$). D'autre part, son diamètre est $2N = \Theta(N)$. Enfin, la distance $\|. - .\|_1$ est compatible avec l'opérateur $\oplus$ défini plus haut.

4.5 Résumé et perspectives

Les réseaux sociaux constituent une gigantesque plateforme sur laquelle il est tentant de réaliser des calculs répartis, tels que des sondages, sur les informations personnelles des utilisateurs. Cependant, à cause du caractère privé des données manipulées et de l'aspect réparti du mode de calcul,

il est difficile de concevoir des protocoles idoines pour ces plateformes, c'est-à-dire extensibles et confidentiels. À ces contraintes s'ajoute la contrainte de précision du calcul. En contrepartie, on dispose d'une caractéristique unique permettant de forcer, de manière coercitive, les participants au calcul à se comporter correctement : un profil, associé à chaque entité du réseau social, qui reflète la personnalité de l'utilisateur réel associé. En supposant que les utilisateurs se soucient de leur réputation, on a pu définir un nouveau modèle de comportement pour les participants au calcul. La définition de ce modèle associé à la spécification du problème de calcul dans un réseau social (le problème S^3) est l'une des deux contributions majeures de ce chapitre. La spécification formelle du problème pose les bases et ouvre la voie d'un nouveau type de calcul réparti. La seconde contribution est la conception de deux protocoles : DPol, un protocole de sondage réparti, qui a permis d'identifier les difficultés du problème ; Agg-S^3 qui constitue un premier pas dans la direction du calcul réparti sécurisé dans les réseaux sociaux. Cependant, un protocole donné n'est qu'un point dans l'espace à cinq dimensions : classe de fonctions calculées, extensibilité, précision, confidentialité et tolérance aux utilisateurs malhonnêtes (f, g, h, faible/fort et $|B|$ dans le problème S^3). Une perspective naturelle de recherche serait de déterminer les frontières de cet espace défini par le problème S^3 et exploré, en parti, lors de la conception de Agg-S^3.

CHAPITRE 5

CONCLUSION

Le travail présenté dans ce manuscrit aborde le problème des utilisateurs malhonnêtes ou tricheurs dans les systèmes répartis à large échelle. La base de ce travail consiste à modéliser les utilisateurs et donc leurs fautes comme étant rationnels, c'est-à-dire ayant un but précis qui dépend de l'application concernée et du contexte, en partant du constat que dans le cas des systèmes collaboratifs répartis l'entité est un utilisateur et non un ordinateur.

L'approche proposée dans ce manuscrit se décompose en plusieurs étapes : (1) modéliser le comportement des utilisateurs en identifiant les choses importantes à leurs yeux, comme le bénéfice qu'un utilisateur tire de son appartenance au système ou un facteur extérieur sur lequel le protocole peut agir de manière directe ou indirecte (par exemple la réputation), (2) concevoir un système de vérifications réparties permettant de détecter les comportements de fautes spécifiés par le modèle d'utilisateur, (3) mettre en œuvre un mécanisme de sanctions capable, d'une part, d'exploiter efficacement les rapports des vérifications et, d'autre part, d'impacter l'intérêt des utilisateurs assurant ainsi de manière dissuasive le bon comportement des tricheurs rationnels.

Les contraintes fixées pour ce travail sont la non-utilisation de la cryptographie, l'extensibilité des solutions proposées et le respect de la confidentialité dans le cas des réseaux sociaux.

Dans le cadre de la diffusion collaborative et épidémique de contenu, le travail réalisé a abouti à la conception de LiFTinG, un protocole léger de détection de tricheurs. Dans ce contexte, le modèle de tricheurs est celui largement utilisé de profiteurs (*freeriders*) et les déviations au protocole se limitent donc à celles permettant soit de diminuer la quantité de donnée envoyée, soit d'augmenter la quantité de donnée reçue. Les principales difficultés rencontrées, outre celles imposées par les contraintes énoncées dans le paragraphe précédent, viennent du fait que les utilisateurs collaborent avec un ensemble dynamique d'autres utilisateurs rendant ainsi les vérifications ardues. Le problème de coalition entre les profiteurs rend cette tâche encore plus difficile. La solution proposée exploite en fait l'aspect aléatoire des protocoles épidémiques pour réduire les risques que les tricheurs se couvrent mutuellement. Les collaborateurs d'un utilisateur vérifient que ce dernier suit la partie déterministe du protocole et émettent des blâmes à son encontre, si besoin est, qu'ils transmettent aux utilisateurs responsables de sa surveillance. En vérifiant de manière statistique que les utilisateurs

choisissent les utilisateurs avec qui ils collaborent de manière uniforme dans l'ensemble du système, on garantit que la majorité des vérifications sera exécutée de manière honnête, conduisant donc à une détection efficace. La sanction utilisée dans ce travail est l'exclusion du système qui réduit à zéro le bénéfice d'un utilisateur. Cependant, le problème de décision permettant de déterminer si un utilisateur doit être exclu ou pas reste une tâche compliquée dans le cas où les pertes de messages et l'aspect aléatoire du protocole peuvent conduire à des blâmes injustes et doit également résister aux tricheries. Dans LiFTinG ce problème est résolu à l'aide d'une analyse théorique permettant de définir et paramétrer de manière rigoureuse la détection.

Le cas du calcul dans les réseaux sociaux, illustré dans un premier temps par la conception de DPol, un algorithme de sondage, est abordé de la même manière au moyen de vérifications réparties. Cependant, contrairement à la première partie de ces travaux, on utilise un facteur extérieur spécifique aux réseaux sociaux en ligne : la réputation des utilisateurs qui peut être impactée en annotant leur profil visible par les autres utilisateurs. Le problème du calcul est alors défini de manière formelle sous la forme du problème S^3 : le calcul sécurisé dans un réseau social, qui doit satisfaire à la fois à des contraintes d'extensibilité, de précision et de confidentialité dans le modèle où les tricheurs tiennent à leur réputation. Ce travail a également conduit à la conception du protocole Agg-S^3 permettant de résoudre le problème S^3 sous certaines conditions de régularité pour la fonction calculée. L'extensibilité du protocole repose sur une structure de groupe facilitant une diffusion de l'information et des vérifications efficaces. La confidentialité est assurée de manière probabiliste à l'aide d'une méthode générique de partage de secret par homomorphisme. La précision repose sur les propriétés de régularité de la fonction calculée exploitées par les vérifications pour limiter les tricheries des utilisateurs malhonnêtes. Ce travail constitue un premier pas dans la direction du calcul réparti centré sur l'utilisateur dans le contexte des réseaux sociaux en ligne.

Au delà des perspectives présentées dans les chapitres précédents, les jeux en ligne massivement multi-joueurs (MMOG pour *Massively Multi-player Online Games*) semblent être un domaine de recherche pouvant bénéficier de l'approche menée pendant mon doctorat et présentée dans ce manuscrit. Pour des raisons d'extensibilité, les MMOG tendent progressivement à être mis en œuvre de manière collaborative ouvrant la possibilité aux utilisateurs malhonnêtes de tricher, ces derniers participant activement au déroulement du jeu. On peut, par exemple, imaginer un utilisateur se cachant des autres joueurs en faisant en sorte que les informations sur sa position ne leur parviennent pas, mettant ainsi en œuvre une attaque furtive ou encore un utilisateur augmentant les capacités de son personnage en altérant les informations le concernant. D'autre part, et c'est là que les méthodes mises en œuvre dans la seconde partie de ce manuscrit interviennent, les éditeurs de ces jeux tendent à lier les personnages à l'identité réelle des utilisateurs à la manière d'un réseau social dans le but d'éviter les tricheries (attaques de Sybil par exemple). Les relations entre utilisateurs sont cependant plus complexes : qui est géographiquement proche de qui dans le jeu, qui est ami avec qui, qui joue régulièrement en même temps que qui, qui et qui s'entraident, *etc.* Enfin, le nombre et la variété des attaques possibles dans le contexte des jeux, dont les règles peuvent être spécifiées de manière formelle sont plus importants que pour les protocoles étudiés dans ce manuscrit. Ceci pousserait à utiliser des méthodes plus sophistiquées de la théorie des jeux et de la vérification de modèle (*model checking* en anglais), ce qui constitue une piste de recherche prometteuse et digne d'intérêt.

BIBLIOGRAPHIE

[AAC+05a] Amitanand AIYER, Lorenzo ALVISI, Allen CLEMENT, Mike DAHLIN, Jean-Philippe MARTIN et Carl PORTH : BAR Fault Tolerance for Cooperative Services. *Operating System Review*, 39:45–58, 2005.

[AAC+05b] Dana ANGLUIN, James ASPNES, Melody CHAN, Michael FISCHER, Hong JIANG et René PERALTA : Stably Computable Properties of Network Graphs. *In DCOSS'05: Proceedings of the 1st International Conference on Distributed Computing in Sensor Systems*, 2005.

[AAD+06] Dana ANGLUIN, James ASPNES, Zoë DIAMADI, Michael FISCHER et René PERALTA : Computation in Networks of Passively Mobile Finite-state Sensors. *Distributed Computing*, 4:235–253, 2006.

[AAER07] Dana ANGLUIN, James ASPNES, David EISENSTAT et Eric RUPPERT : The Computational Power of Population Protocols. *Distributed Computing*, 20:279–304, 2007.

[AH00] Eytan ADAR et Bernardo HUBERMAN : Free Riding on Gnutella. *First Monday*, 5:2–13, 2000.

[AL86] Algirdas AVIŽIENIS et Jean-Claude LAPRIE : Dependable Computing: From Concepts to Design Diversity. *Proceedings of the IEEE*, 74:629–638, 1986.

[Bat] `http://www.battle.net`.

[BDHU09] Michael BACKES, Peter DRUSCHEL, Andreas HAEBERLEN et Dominique UNRUH : CSAR: A Practical and Provable Technique to Make Randomized Systems Accountable. *In NDSS'09: Proceedings of the 16th Annual Network & Distributed System Security Symposium*, 2009.

[Ben86] Josh BENALOH : Secret Sharing Homomorphisms: Keeping Shares of a Secret Secret. *In CRYPTO'86: Proceedings of the 6th Annual International Conference on Advances in Cryptology*, 1986.

[Ben94] Josh BENALOH : Dense Probabilistic Encryption. *In SAC'94: In Proceedings of the 1st Workshop on Selected Areas of Cryptography*, 1994.

[BGK+09] Edward BORTNIKOV, Maxim GUREVICH, Idit KEIDAR, Gabriel KLIOT et Alexander SHRAER : Brahms: Byzantine Resilient Random Membership Sampling. *Computer Networks*, 53:2340–2359, 2009.

[CGMA85] Benny CHOR, Shafi GOLDWASSER, Silvio MICALI et Baruch AWERBUCH : Verifiable Secret Sharing and AchievingSsimultaneity in the Presence of Faults. *In FOCS*

'85: Proceedings of the 26th Annual Symposium on Foundations of Computer Science, 1985.

[CL02] Miguel CASTRO et Barbara LISKOV : Practical Byzantine Fault Tolerance and Proactive Recovery. *ACM Transactions on Computer Systems*, 20:398–461, 2002.

[CN03] Landon COX et Brian NOBLE : Samsara: Honor Among Thieves in Peer-to-peer Storage. *In SOSP'03: Proceedings of 17th ACM Symposium on Operating Systems Principles*, 2003.

[Coh03] Bram COHEN : Incentives Build Robustness in BitTorrent. *In P2P Econ'03: Proceedings of the 1st Workshop on the Economics of Peer-to-Peer Systems*, 2003.

[CT91] Thomas COVER et Joy THOMAS : *Elements of Information Theory*. John Wiley and Sons, Inc., 1991.

[CWA+09] Allen CLEMENT, Edmund WONG, Lorenzo ALVISI, Mike DAHLIN et Mirco MARCHETTI : Making Byzantine Fault Tolerant Systems Tolerate Byzantine Faults. *In NSDI'09: Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation*, 2009.

[DGFGR06] Carole DELPORTE-GALLET, Hugues FAUCONNIER, Rachid GUERRAOUI et Eric RUPPERT : When Birds Die: Making Population Protocols Fault-tolerant. *In DCOSS'06: Proceedings of the 2nd International Conference on Distributed Computing in Sensor Systems*, 2006.

[DGFGR07] Carole DELPORTE-GALLET, Hugues FAUCONNIER, Rachid GUERRAOUI et Eric RUPPERT : Secretive Birds: Privacy in Population Protocols. *In OPODIS'07: Proceedings of the 11th International Conference on Principles of Distributed Systems*, 2007.

[Dig] `http://digg.com/`.

[DIK+08] Ivan DAMGÅRD, Yuval ISHAI, Mikkel KRØIGAARD, Jesper Buus NIELSEN et Adam SMITH : Scalable Multiparty Computation with Nearly Optimal Work and Resilience. *In CRYPTO'08: Proceedings of the 28th Annual International Conference on Advances in Cryptology*, 2008.

[DN07] Ivan DAMGÅRD et Jesper Buus NIELSEN : Scalable and Unconditionally Secure Multi-Party Computation. *In CRYPTO'07: Proceedings of the 27th Annual International Conference on Advances in Cryptology*, 2007.

[Doo10] DOODLE AG : Facebook Doodle. Online, 2010. `http://www.facebook.com/apps/application.php?id=6314677796`.

[Dou02] John DOUCEUR : The Sybil Attack. *In IPTPS'02: Proceedings of the First International Workshop on Peer-to-Peer Systems*, 2002.

[Dwo06] Cynthia DWORK : Differential Privacy. *In ICALP'06: Proceedings of the 33rd International Colloquium on Automata, Languages and Programming*, 2006.

[DXL+06] Mayur DESHPANDE, Bo XING, Iosif LAZARDIS, Bijit HORE, Nalini VENKATASUBRAMANIAN et Sharad MEHROTRA : CREW: A Gossip-based Flash-Dissemination System. *In ICDCS'06: Proceedings of the 26th IEEE International Conference on Distributed Computing Systems*, 2006.

[EGH+03] Patrick Th. EUGSTER, Rachid GUERRAOUI, Sidath HANDURUKANDE, Petr KOUZNETSOV et Anne-Marie KERMARREC : Lightweight Probabilistic Broadcast. *ACM Transactions on Computer Systems*, 21(4):341–374, 2003.

[Fac] `http://www.facebook.com/`.

[FCC+03] Yun FU, Jeffrey CHASE, Brent CHUN, Stephen SCHWAB et Amin VAHDAT : SHARP: An Architecture for Secure Resource Peering. *In SOSP'03: Proceedings of 17th ACM Symposium on Operating Systems Principles*, 2003.

[FGK+09a] Davide FREY, Rachid GUERRAOUI, Anne-Marie KERMARREC, Boris KOLDEHOFE, Martin MOGENSEN, Maxime MONOD et Vivien QUÉMA : Heterogeneous Gossiping. *In MIDDLEWARE'09: Proceedings of the 10th ACM/IFIP/USENIX International Middleware Conference*, 2009.

[FGK+09b] Davide FREY, Rachid GUERRAOUI, Anne-Marie KERMARREC, Maxime MONOD et Vivien QUÉMA : Stretching Gossip with Live Streaming. *In DSN'09: Proceedings of the 39th IEEE/IFIP International Conference on Dependable Systems and Networks*, 2009.

[FM10] Davide FREY et Maxime MONOD : GossipLib. Online, 2010. `http://gossiplib.gforge.inria.fr/`.

[GBL+03] Indranil GUPTA, Kenneth BIRMAN, Prakash LINGA, Alan DEMERS et Robbert van RENESSE : Kelips: Building an Efficient and Stable P2P DHT through Increased Memory and Background Overhead. *In IPTPS'03: Proceedings of the Second International Workshop on Peer-to-Peer Systems*, 2003.

[GGHK10] Andrei GIURGIU, Rachid GUERRAOUI, Kévin HUGUENIN et Anne-Marie KERMARREC : Computing in Social Networks. *In SSS'10: Proceedings of the 12th International Symposium on Stabilization, Safety and Security of Distributed Systems*, 2010.

[GHK+10] Rachid GUERRAOUI, Kévin HUGUENIN, Anne-Marie KERMARREC, Maxime MONOD et Swagatika PRUSTY : LiFTinG: Lightweight Freerider-Tracking Protocol in Gossip. *In MIDDLEWARE'10: Proceedings of the 11th ACM/IFIP/USENIX International Middleware Conference*, 2010.

[GHKM09a] Rachid GUERRAOUI, Kévin HUGUENIN, Anne-Marie KERMARREC et Maxime MONOD : Decentralized Polling with Respectable Participants. *In OPODIS'09: Proceedings of the 12th International Conference on Principles of Distributed Systems*, 2009.

[GHKM09b] Rachid GUERRAOUI, Kévin HUGUENIN, Anne-Marie KERMARREC et Maxime MONOD : On Tracking Freeriders in Gossip Protocols (Short paper). *In P2P'09: Proceedings of the 9th IEEE International Conference on Peer to Peer Computing*, 2009.

[GHKM09c] Rachid GUERRAOUI, Kévin HUGUENIN, Anne-Marie KERMARREC et Maxime MONOD : Towards Secured Distributed Polling in Social Networks. *In DISC'09: Proceedings of the 23rd International Symposium on Distributed Computing*, 2009.

[GKM01] Ayalvadi GANESH, Anne-Marie KERMARREC et Laurent MASSOULIÉ : SCAMP: Peer-to-peer Lightweight Membership Service for Large-scale Group Communication. *In NGC'01: Proceedings of the Third International Workshop on Networked Group Communication*, 2001.

[GKQV10] Rachid GUERRAOUI, Nikola KNEŽEVIĆ, Vivien QUÉMA et Marko VUKOLIĆ : The Next 700 BFT Protocols. *In EuroSys'10: Proceedings of the 5th European conference on Computer Systems*, 2010.

[GR09] Rachid GUERRAOUI et Eric RUPPERT : Names Trump Malice: Tiny Mobile Agents Can Tolerate Byzantine Failures. *In ICALP'09: Proceedings of the 36st International Colloquium on Automata, Languages and Programming*, 2009.

[GV08] Josh GOODMAN et Clark VERBRUGGE : A Peer Auditing Scheme for Cheat Elimination in MMOGs. *In NetGames'08: Proceedings of the 7th Annual Workshop on Network and Systems Support for Games*, 2008.

[GY87] Zvi GALIL et Moti YUNG : Partitioned Encryption and Achieving Simultaneity by Partitioning. *Information Processing Letters*, 26:81–88, 1987.

[HCW05] Daniel HUGHES, Geoff COULSON et James WALKERDINE : Free Riding on Gnutella Revisited: The Bell Tolls? *IEEE Distributed Systems Online*, 6:1, 2005.

[HJPVR08] Maya HARIDASAN, Ingrid JANSCH-PORTO et Robbert VAN RENESSE : Enforcing Fairness in a Live-Streaming System. *In MMCN'08: Proceedings of the Fifteenth Annual Multimedia Computing and Networking*, 2008.

[HKD07] Andreas HAEBERLEN, Petr KOUZNETSOV et Peter DRUSCHEL : PeerReview: Practical Accountability for Distributed Systems. *In SOSP'07: Proceedings of 21st ACM Symposium on Operating Systems Principles*, 2007.

[JGKvS04] Márk JELASITY, Rachid GUERRAOUI, Anne-Marie KERMARREC et Maarten van STEEN : The Peer Sampling Service: Experimental Evaluation of Unstructured Gossip-Based Implementations. *In MIDDLEWARE'04: Proceedings of the 5th ACM/IFIP/USENIX International Conference on Middleware*, 2004.

[JMB04] Márk JELASITY, Alberto MONTRESOR et Ozalp BABAOGLU : Detection and Removal of Malicious Peers in Gossip-Based Protocols. *In FuDiCo II (S.O.S.): Proceedings of the 2nd Bertinoro Workshop on Future Directions in Distributed Computing: Survivability: Obstacles and Solutions*, 2004.

[JVG+07] Márk JELASITY, Spyros VOULGARIS, Rachid GUERRAOUI, Anne-Marie KERMARREC et Maarten van STEEN : Gossip-based Peer Sampling. *ACM Transactions on Computer Systems*, 25:1–36, 2007.

[KAD+09] Ramakrishna KOTLA, Lorenzo ALVISI, Michael DAHLIN, Allen CLEMENT et Edmund WONG : Zyzzyva: Speculative Byzantine Fault Tolerance. *ACM Transactions on Computer Systems*, 27:398–461, 2009.

[KKU08] Murat KARAKAYA, İbrahim KÖRPEOĞLU et Özgür ULUSOY : Counteracting Free-riding in Peer-to-Peer Networks. *Computer Networks*, 52:675–694, 2008.

[KMG03] Anne-Marie KERMARREC, Laurent MASSOULIÉ et Ayalvadi GANESH : Probabilistic Reliable Dissemination in Large-Scale Systems. *IEEE Transactions on Parallel and Distributed Systems*, 14:248–258, 2003.

[KPG+05] Dionysios KOSTOULAS, Dimitrios PSALTOULIS, Indranil GUPTA, Ken BIRMAN et Alan DEMERS : Decentralized schemes for size estimation in large and dynamic groups. *In NCA'05: Proceedings of the 4th IEEE International Symposium on Network Computing and Applications*, 2005.

[KPQS09] Anne-Marie KERMARREC, Alessio PACE, Vivien QUÉMA et Valerio SCHIAVONI : NAT-resilient Gossip Peer Sampling. *In ICDCS'09: Proceedings of the 29th IEEE International Conference on Distributed Computing Systems*, 2009.

[Kre10] KREMSA DESIGN, INC. : Facebook Poll. Online, 2010. `http://www.facebook.com/apps/application.php?id=20678178440`.

[KS04] Valerie KING et Jared SAIA : Choosing a Random Peer. *In PODC'04: Proceedings of the Twenty-Third Annual ACM Symposium on Principles of Distributed Computing*, 2004.

[KSTT04] Ramayya KRISHNAN, Michael SMITH, Zhulei TANG et Rahul TELANG : The Impact of Free-Riding on Peer-to-Peer Networks. *In HICSS'04: Proceedings of the 37th Hawaii International Conference on System Sciences*, 2004.

[KTCB05] Patric KABUS, Wesley W. TERPSTRA, Mariano CILIA et Alejandro BUCHMANN : Addressing Cheating in Distributed MMOGs. *In NetGames'05: Proceedings of the 4th Annual Workshop on Network and Systems Support for Games*, 2005.

[LCM+08] Harry LI, Allen CLEMENT, Mirco MARCHETTI, Manos KAPRITSOS, Luke ROBISON, Lorenzo ALVISI et Mike DAHLIN : FlightPath: Obedience vs. Choice in Cooperative Services. *In OSDI'08: Proceedings of the 8th USENIX Symposium on Operating Systems Design and Implementation*, 2008.

[LCW+06] Harry LI, Allen CLEMENT, Edmund WONG, Jeff NAPPER, Indrajit ROY, Lorenzo ALVISI et Michael DAHLIN : BAR Gossip. *In OSDI'06: Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation*, 2006.

[LMSW06] Thomas LOCHER, Patrick MOOR, Stefan SCHMID et Roger WATTENHOFER : Free Riding in BitTorrent is Cheap. *In HotNets V: Proceedings of the Second Workshop on Hot Topics in Networks*, 2006.

[LSP82] Leslie LAMPORT, Robert SHOSTAK et Marshall PEASE : The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, 4:382–401, 1982.

[LXQ+08] Bo LI, Susu XIE, Yang QU, Gabriel KEUNG, Chuang LIN, Jiangchuan LIU et Xinyan ZHANG : Inside the New Coolstreaming: Principles, Measurements and Performance Implications. *In INFOCOM'08: Proccedings of the 27th IEEE Conference on Computer Communications*, 2008.

[Mai98] George J. MAILATH : Do People Play Nash Equilibrium? Lessons from Evolutionary Game Theory. *Journal of Economic Literature*, 36(3):1347–1374, 1998.

[MG09] Ramsés MORALES et Indranil GUPTA : AVMON: Optimal and Scalable Discovery of Consistent Availability Monitoring Overlays for Distributed Systems. *IEEE Transactions on Parallel and Distributed Systems*, 20:446–459, 2009.

[MLMKG06] Laurent MASSOULIÉ, Erwan LE MERRER, Anne-Marie KERMARREC et Ayalvadi GANESH : Peer Counting and Sampling in Overlay Networks: Random Walk Methods. *In PODC'06: Proceedings of the Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing*, 2006.

[MMP02] Dahlia MALKHI, Ofer MARGO et Elan PAVLOV : E-Voting without 'Cryptography'. *In FC'02: Proceedings of the Sixth International Financial Cryptography Conference*, 2002.

[MPGD08] Alan MISLOVE, Ansley POST, Krishna GUMMADI et Peter DRUSCHEL : Ostra: Leveraging Trust to Thwart Unwanted Communication. *In NSDI'08: Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, 2008.

[Nas50] John NASH : Equilibrium Points in n-person Games. *In Proceedings of the National Academy of Sciences of the United States of America*, 1950.

[Nas51] John NASH : Non-cooperative Games. *The Annals of Mathematics*, 54:286–295, 1951.

[NM44] John Von NEUMANN et Oskar MORGENSTERN : *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

[NNS+05] Animesh NANDI, Tsuen-Wan NGAN, Atul SINGH, Peter DRUSCHEL et Dan WALLACH : Scrivener: Providing Incentives in Cooperative Content Distribution Systems. *In MIDDLEWARE'05: Proceedings of the 6th ACM/IFIP/USENIX International Conference on Middleware*, 2005.

[NWD03] Tsuen-Wan NGAN, Dan WALLACH et Peter DRUSCHEL : Enforcing Fair Sharing of Peer-to-peer Resources. *In IPTPS'03: Proceedings of the Second International Workshop on Peer-to-Peer Systems*, 2003.

[Pai99] Pascal PAILLIER : Public-key Cryptosystems Based on Composite Degree Residuosity Classes. *In EUROCRYPT'99: Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques*, 1999.

[PIA+07] Michael PIATEK, Tomas ISDAL, Thomas ANDERSON, Arvind KRISHNAMURTHY et Arun VENKATARAMANI : Do Incentives Build Robustness in BitTorrent? *In NSDI'07: Proceedings of the 4th USENIX Symposium on Networked Systems Design and Implementation*, 2007.

[Pok] `http://www.pokerstars.com`.

[RAD78] R. RIVEST, L. ADLEMAN et M. DERTOUZOS : On Data Banks and Privacy Homomorphisms. *Foundations of Secure Computation*, pages 169–177, 1978.

[RBO89] Tal RABIN et Michael BEN-OR : Verifiable Secret Sharing and Multi-Party Protocols with Honest Majority. *In STOC '89: Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, 1989.

[RS07] Ronald RIVEST et Warren SMITH : Three Voting Protocols: ThreeBallot, VAV, and Twin. *In EVT'07: Proceedings of the 2007 USENIX/ACCURATE Electronic Voting Technology Workshop*, 2007.

[RSK+10] Indrajit ROY, Srinath SETTY, Ann KILZER, Vitaly SHMATIKOV et Emmett WITCHEL : Airavat: Security and Privacy for MapReduce. *In NSDI'10: Proceedings of the 7th USENIX Symposium on Networked Systems Design and Implementation*, 2010.

[Sam01] Pierangela SAMARATI : Protecting Respondents' Identities in Microdata Release. *IEEE Transactions on Knowledge and Data Engineering*, 13:1010–1027, 2001.

[Sch99] Berry SCHOENMAKERS : A Simple Publicly Verifiable Secret Sharing Scheme and its Application to Electronic Voting. *In CRYPTO'99: Proceedings of the 19th Annual International Conference on Advances in Cryptology*, 1999.

[Sha79] Adi SHAMIR : How to Share a Secret. *Communications of the ACM*, 22:612–613, 1979.

[SKY10] Michael SIRIVIANOS, Kyungbaek KIM, Kim et Xiaowei YANG : SocialFilter: Introducing Social Trust to Collaborative Spam Mitigation. *In ColSec'10: Proceedings of the 2010 Usenix Workshop on Collaborative Methods for Security and Privacy*, 2010.

[SPCY07] Michael SIRIVIANOS, Jong PARK, Rex CHEN et Xiaowei YANG : Free-riding in BitTorrent with the Large View Exploit. *In IPTPS'07: Proceedings of the 6th International Workshop on Peer-to-Peer Systems*, 2007.

[SPYJ07] Michael SIRIVIANOS, Jong Han PARK, Xiaowei YANG et Stanislaw JARECKI : Dandelion: Cooperative Content Distribution with Robust Incentives. *In ATC'07: Proceedings of the 2007 USENIX Annual Technical Conference*, 2007.

[Ste74] M. A. STEPHENS : EDF Statistics for Goodness of Fit and Some Comparisons. *Journal of the American Statistical Association*, 69:730–737, 1974.

[Swe02] Latanya SWEENEY : k-Anonymity : A Model for Protecting Privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10:557–570, 2002.

[TMLS09] Nguyen TRAN, Bonan MIN, Jinyang LI et Lakshminarayanan SUBRAMANIAN : Sybil-Resilient Online Content Voting. *In NSDI'09: Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation*, 2009.

[VABD09] Le-Hung VU, Karl ABERER, Sonja BUCHEGGER et Anwitaman DATTA : Enabling Secure Secret Sharing in Distributed Online Social Networks. *In ACSAC 25: Proceedings of the 2009 Annual Computer Security Applications Conference*, pages 419–428, 2009.

[VGvS05] Syros VOULGARIS, Daniela GAVIDIA et Maarten van STEEN : CYCLON: Inexpensive Membership Management for Unstructured P2P Overlays. *Journal of Network and Systems Management*, 13:197–217, 2005.

[VYF06] Vidhyashankar VENKATARAMAN, Kaouru YOSHIDA et Paul FRANCIS : ChunkySpread: Heterogeneous Unstructured Tree-Based Peer-to-Peer Multicast. *In ICNP'06: Proceedings of the 14th IEEE International Conference on Network Protocols*, 2006.

[Yao82] Andrew YAO : Protocols for Secure Computations. *In FOCS '82: Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, 1982.

[YGKX08] Haifeng YU, Phillip GIBBONS, Michael KAMINSKY et Feng XIAO : SybilLimit: A Near-Optimal Social Network Defense against Sybil Attacks. *In SP'08: Proceedings of the 2008 IEEE Symposium on Security and Privacy*, 2008.

[YKGF08] Haifeng YU, Michael KAMINSKY, Phillip GIBBONS et Abraham FLAXMAN : SybilGuard: Defending Against Sybil Attacks via Social Networks. *IEEE/ACM Transactions on Networking*, 16:576–589, 2008.

[ZZSY07] Meng ZHANG, Qian ZHANG, Lifeng SUN et Shiqiang YANG : Understanding the Power of Pull-Based Streaming Protocol: Can We Do Better? *IEEE Journal on Selected Areas in Communications*, 25:1678–1694, 2007.

Résumé

Techniquement parlant, Internet est un système réparti constitué d'ordinateurs et par conséquent les modèles de fautes couramment utilisés prennent principalement en compte les problèmes matériels : pannes, pertes de messages, *etc*. À l'autre extrême, le modèle de fautes byzantines inclut tous les comportements possibles. Dans le cas de nombreux services répartis déployés sur Internet, des systèmes de partage de fichiers aux réseaux sociaux, une entité du système est un utilisateur et non un ordinateur. Par conséquent, la définition d'un nouveau modèle de fautes centré sur l'utilisateur s'impose : celui du *tricheur rationnel*, c'est-à-dire celui d'un utilisateur prêt à dévier du protocole, mais de manière sensée, pour servir un intérêt précis.

Le travail présenté dans ce manuscrit s'appuie sur le modèle de tricheurs rationnels et propose une méthode dissuasive utilisant des vérifications réparties extensibles pour détecter les fautes et ainsi lutter contre les tricheurs, et ce, sans recourir à la cryptographie. Dans un premier temps, on aborde le cas des systèmes collaboratifs de dissémination épidémique de contenu et on montre que les profiteurs, c'est-à-dire des utilisateurs tentant de maximiser leur bénéfice tout en minimisant leur contribution, peuvent être détectés efficacement par un mécanisme de surveillance mutuelle en exploitant l'aspect aléatoire du système. Dans un second temps, on s'intéresse au contexte des réseaux sociaux où le lien explicite entre un utilisateur et une personne réelle permet d'utiliser la réputation de cette dernière comme moyen de dissuasion. On définit de cette manière un modèle de fautes pour un utilisateur d'un réseau social et on formule le problème de calcul extensible et sécurisé, c'est-à-dire précis et confidentiel, que l'on résout à l'aide d'un protocole également présenté dans ce manuscrit.

Abstract

From a technical point of view, the Internet is a distributed system of computers. As such, traditional fault models mainly take into account hardware faults such as fail stop and message losses. On the other hand, the Byzantine fault model encompasses all possible behaviours. In most distributed services deployed over the Internet, ranging from collaborative file sharing to social networks, the basic entity is a user, not a computer. Therefore, one needs a new user-centric fault model: a model of rational cheaters, *i.e.*, is users who deviate from the protocol to serve a given interest in a sensible way.

The work presented in this manuscript relies on a model of rational cheaters and proposes a dissuasive approach based on scalable distributed verifications to detect faults and fight against cheaters without the need for cryptography. First we consider the context of large-scale high-bandwidth content dissemination in gossip and we show that freeriders, that is users that maximize their benefits while minimizing their contributions, can be efficiently detected using a distributed monitoring system that leverages the inherent randomness of gossip. Second, we focus on social networks in which the reputation concern coming from the one-to-one mapping between a user and a real person can be exploited as an incentive to well behave. Based on this defining characteristic, we devise a new model for users of social networks and we formalize the problem of scalable and secure computing, that is accurate and private, and propose a protocol that solves it.