Système de réputation préservant la vie privée

Paul Lajoie-Mazenc

UNIVERSITÉ RENNES-1 — SUPÉLEC

RAPPORT DE STAGE

Systèmes de réputation préservant la vie privée

EPC CIDER / EPC CIDre

Stagiaire :
Paul LAJOIE-MAZENC

Encadrants :
Emmanuelle ANCEAUME
Gilles GUETTE
Nicolas PRIGENT
Valérie VIET TRIEM TONG

Master Recherche en Informatique

Printemps 2012

Résumé

Un système de réputation est un système résumant le comportement passé d'entités, permettant de savoir si celles-ci sont dignes de confiance pour des échanges futurs.

Dans ce rapport de stage, nous proposons un système de réputation robuste présentant certaines propriétés de respect de la vie privée intéressantes. Nous décrivons l'état de l'art des systèmes de réputation, définissons formellement les propriétés recherchées et présentons notre proposition, en expliquant pourquoi les propriétés sont respectées.

Mots-clés Systèmes distribués à grande échelle – Systèmes de réputation – Respect de la vie privée

Table des matières

Chapitre 1

Introduction

Dans les grands réseaux comme Internet, la grande majorité des interactions sont effectuées entre des inconnus, ce qui pose problème lorsque les membres du réseau placent dans ces interactions une certaine valeur, monétaire ou autre. Dans le cas du commerce électronique par exemple, un acheteur n'a aucune idée de l'état réel du bien vendu, qui peut être un produit d'occasion au lieu d'un produit neuf ou être plus abîmé qu'annoncé. Parallèlement, le vendeur ne peut pas être certain qu'il sera payé après avoir expédié le bien. De ce fait, l'acheteur et le vendeur aimeraient tous deux savoir avant de s'engager définitivement s'ils peuvent faire confiance à l'autre et si le risque qu'ils encourent est grand.

Pour répondre à ce problème, il est possible d'utiliser un système de réputation. Un système de réputation permet à ses utilisateurs d'estimer la réputation des autres utilisateurs afin de les aider à décider si oui ou non ils peuvent leur faire confiance et s'il est prudent de conclure une transaction. La réputation (ou *score de réputation*) d'un membre du réseau est généralement représentée par un objet mathématique. L'objectif du score de réputation est de donner une représentation synthétique des différents avis des clients ayant déjà eu une interaction avec le fournisseur. Dans le cas du système de commerce électronique eBay [1], un vendeur propose un objet à la vente en fournissant une description ainsi qu'une photographie. Les acheteurs potentiels vont enchérir afin de remporter cet objet. Une fois que la transaction est effectuée, l'acheteur et le vendeur peuvent se noter l'un l'autre en déposant un *témoignage*, c'est-à-dire une note dans $\{-1, 0, +1\}$. Le score de réputation d'un utilisateur est la somme de ces retours. Il existe d'autres méthodes de calcul de score, comme l'utilisation de fonctions bayésiennes [18].

Si personne ne cherche à contourner le système, tout ira pour le mieux. Cependant, certains utilisateurs peuvent essayer d'attaquer le système, par exemple pour augmenter leur réputation, la remettre à zéro ou diminuer celle d'autres utilisateurs. D'autres peuvent adopter un comportement égoïste et faire du *free-riding*, c'est-à-dire utiliser le système pour estimer la réputation de certains utilisateurs sans fournir eux-mêmes de témoignages. Comme nous le verrons par la suite, des solutions existent permettant de se prémunir de telles attaques. Ces solutions utilisent généralement des techniques de redondance des informations, des fonctions de calcul de réputation robustes ou encore des mécanismes d'incitation à la coopération.

Par contre, très rares sont les solutions qui, au-delà d'être robustes, garantissent aux utilisateurs un certain respect de leur vie privée. C'est cette problématique qui motive ce travail. Pour cela, nous considérons le cas d'un réseau dans lequel des *fournisseurs de services* fournissent des services aux autres membres du réseaux que nous appelons les *clients*. Nous proposons un système de réputation permettant aux clients de décider s'ils peuvent en toute confiance interagir avec un fournisseur de service ou non. Ce système de réputation est *distribué* afin qu'il n'existe pas de point unique de défaillance sur lequel un attaquant puisse focaliser ses efforts. De plus, notre proposition préserve la vie privée des clients grâce à l'utilisation de pseudonymes.

Le Chapitre 2 présente l'état de l'art des systèmes de réputation et un exemple de système de réputation centralisé préservant la vie privée de ses participants. Le Chapitre 3 définit formellement les termes utilisés et explique quels sont nos objectifs. Le Chapitre 4 présente notre proposition, en présentant d'abord un système de réputation naïf puis en l'améliorant et en détaillant le protocole d'interaction entre ses participants. Le Chapitre 5 présente les attaques que ce système permet d'éviter, et explique en quoi notre proposition respecte la vie privée des clients. Finalement, nous concluons dans le Chapitre 6 et présentons les principaux travaux futurs.

Chapitre 2

État de l'art

Avant de proposer notre système de réputation, nous présentons l'état de l'art des systèmes de réputation existants, qu'ils soient centralisés ou distribués. Dans un premier temps, nous présentons la terminologie commune à tous les systèmes de réputation. Celle-ci sera enrichie dans la Section 3.1.

2.1 Terminologie des systèmes de réputation

Un système de réputation met en jeu des participants appelés *agents*. Un agent peut être soit un *fournisseur de service*, soit un *client*. Un fournisseur de service, noté FS dans la suite, fournit un service, par un exemple un bien dans le cas d'un système de commerce électronique ou un fichier dans un réseau P2P de transfert de fichiers. Ces fournisseurs sont contactés par des clients qui utilisent les services proposés. Les clients sont notés $p, q, \ldots$ Avant d'interagir avec un fournisseur, un client utilise le système de réputation pour calculer le *score de réputation* du fournisseur de service, lui permettant ainsi d'estimer le comportement passé de ce fournisseur et de décider si oui ou non il peut interagir avec lui sans risque. Une fois que l'interaction est finie, le client émet un *témoignage* décrivant ce qu'il pense de cette transaction. Les nœuds du réseau stockant les témoignages sont appelés des *témoins*.

Bien évidemment, les différents agents peuvent être *bienveillants* ou *malveillants*. Un agent malveillant va par exemple essayer d'augmenter artificiellement un score de réputation ou en diminuer un autre.

2.2 Systèmes de réputation

Lorsqu'un client utilise un système de réputation, il doit effectuer plusieurs actions pour obtenir le score de réputation d'un fournisseur de service. Dans un premier temps, il doit localiser où sont *stockés* les témoignages concernant les actions passées de ce fournisseur. Si le système est distribué, les témoignages peuvent être répartis dans le réseau. Ceux-ci peuvent-être les agents ayant émis ces témoignages ou d'autres. Le client doit alors déterminer en quels endroits collecter les témoignages, c'est-à-dire *localiser les témoins* de ce fournisseur. Ensuite, il doit les *collecter* le plus efficacement possible. Finalement, il doit *calculer le score de réputation* du fournisseur. Ces trois étapes –

quatre dans le cas distribué – sont les briques communes à tout système de réputation.

À cet effet, nous proposons une taxonomie des systèmes de réputation suivant ces quatre éléments. Nous ferons ensuite le point sur les attaques possibles sur les systèmes de réputation. Finalement, nous présenterons un système de réputation centralisé respectant la vie privée des participants.

2.2.1 Stockage des témoignages

Choisir une architecture centralisée ou distribuée modifie le stockage des informations. En effet, lorsqu'un serveur central est utilisé par tous les clients et fournisseurs de service, toutes les informations peuvent y être stockées.

Par contre, lorsque la conception du système de réputation est distribuée, les témoignages peuvent être stockés par l'agent comme proposé par Ravoaja [28], sous l'hypothèse que celui-ci n'a aucun intérêt à modifier son propre retour[1]. C'est ce qui est proposé dans la plupart des systèmes distribués [20] [25].

2.2.2 Sélection des témoins dans une architecture distribuée

En l'absence de serveur central sur lequel l'intégralité des témoignages concernant un fournisseur de service est stockée, il est nécessaire de localiser les témoignages. Comme le réseau est généralement ouvert, il peut y avoir une grande quantité de nœuds sur lesquels les témoignages ont été stockés à l'issue des différentes interactions entre clients et fournisseurs de service. Interroger tous ces nœuds est trop coûteux, c'est pourquoi des schémas de sélection de témoins sont utilisés.

Ravoaja [28] présente plusieurs méthodes de sélection des témoins. La première, naïve, est une méthode par inondation. Un client cherchant à calculer la réputation de FS demande à chacun de ses voisins à un saut sur le réseau s'il a interagi avec FS. Ces derniers transmettent la requête à leurs propres voisins et ainsi de suite, jusqu'à ce que le nombre de sauts maximum (ou Time To Live), donné dans le premier message, soit atteint.

Une première optimisation consiste en l'utilisation de marches aléatoires. Le premier témoin est choisi de manière aléatoire et choisira ensuite un autre témoin aléatoirement parmi ses propres témoins, et ainsi de suite jusqu'à ce que le nombre désiré de témoins soit obtenu.

Une seconde technique possible est celle dite de « recherche par percolation ». Pour cette méthode, chaque témoin initie une marche aléatoire de longueur fixée et donne un index pointant sur ses témoignages à chaque agent de la marche aléatoire. Si, à l'instar de Gnutella, le réseau suit une distribution en loi de puissance[2], Ravoaja montre qu'au moins un agent ayant une réputation élevée reçoit une copie de l'index si la longueur de la marche est suffisante. Cela permet une propagation très rapide des requêtes et assure aux clients qu'en contactant les agents de score élevé ils auront des pointeurs vers la plupart des témoignages.

[1] Cette hypothèse paraît néanmoins trompeuse dans le sens où un agent pourrait vouloir semer la zizanie entre d'autres agents.

[2] C'est-à-dire que peu d'agents ont une réputation élevée, tandis que la plupart ont une réputation relativement faible.

Enfin, le réseau sous-jacent peut être structuré par une DHT – *Distributed Hash Table*. Cette structure régit la position des agents et des ressources dans le réseau. Dans le cas de Chord [32], 2^N nœuds et ressources du système sont organisés en anneau en fonction du haché d'un identifiant : adresse IP, nom, etc. Chaque nœud ou ressource est identifié par ce haché, qui donne un nombre entre 1 et 2^N. Afin de communiquer efficacement avec les autres nœuds, chacun d'entre eux possède une table de routage (*finger table*) lui permettant de communiquer directement avec N autres nœuds : n_i peut communiquer avec n_{i+1} et n_{i+2^j} pour $j \in \{0, \ldots, N-1\}$, comme présenté à la figure 2.1 pour $N = 6$ et 9 nœuds. Si un nœud ne peut être atteint directement, les messages sont transmis de proche en proche.

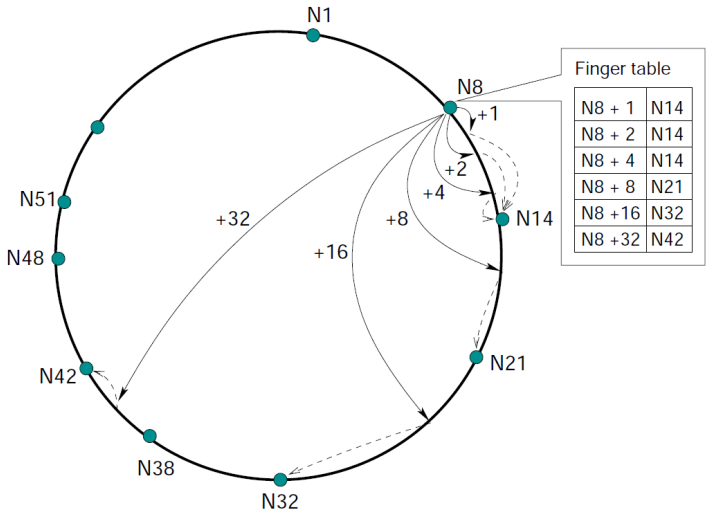


FIG. 2.1: Exemple de réseau Chord [32]

2.2.3 Collecte des témoignages

Dans toute architecture, la manière dont les scores de réputation du serveur sont collectés peut varier.

Dans le cas centralisé, il suffit que le serveur donne le score global de réputation d'un fournisseur de service à tout agent en faisant la requête. C'est ce qui est fait pour eBay : tout le monde peut voir sur le profil d'un vendeur quel est son score de réputation et quelles sont ses dernières évaluations.

Dans une architecture distribuée, la collecte des témoignages est fortement liée à l'algorithme de stockage des témoignages : par exemple, si les témoignages sont stockés sur les agents les ayant émis, il faut localiser ces agents témoins via une des méthodes présentées précédemment puis les contacter.

Pavlov *et al.* [25] proposent plusieurs méthodes permettant en outre une collecte anonyme des témoignages. Dans celles-ci, le paramètre n représente le nombre de témoins de FS choisis par p.

Dans la première méthode, p construit un anneau constitué de tous les témoins. Pour calculer le score de réputation d'un fournisseur de service, p

fait transiter un nombre qu'il initialise à une valeur aléatoire. Lors de la réception du message, chaque témoin additionne à ce nombre son témoignage sur FS et envoie le résultat au témoin suivant. Lorsque p obtient la somme de toutes les notes, il soustrait la valeur aléatoire initiale pour obtenir *un* score de réputation de FS – ce score dépendant des témoins choisis. Au total, $O(n)$ messages ont été envoyés entre les agents. Le problème est que deux témoins « entourant » un autre témoin peuvent obtenir le score du témoin entouré. Pour améliorer cela, les auteurs proposent une deuxième méthode.

Dans celle-ci, les témoins sont organisés dans un graphe complet, c'est-à-dire que chacun connaît tous les autres témoins. p choisit ensuite une valeur r_p aléatoirement. Chaque témoin va alors séparer son secret – son témoignage sur FS – en $n+1$ parties $r_{i,1}, \ldots, r_{i,n+1}$, p étant vu comme le $n+1$-ième témoin. Chacun en envoie ensuite une partie à chaque autre agent, et en conserve une. Lors de la réception du message, chaque agent calcule la somme des valeurs reçues et l'envoie à p. Il somme ces valeurs et soustrait r_p de la somme obtenue. En tout, ce protocole requiert $O(n^2)$ messages. Ce protocole améliore la résilience du système face aux agents malveillants.

2.2.4 Calcul du score de réputation

Le calcul du score de réputation est intimement lié au format des témoignages, et donc à leur propagation ainsi qu'à leur stockage. Jøsang *et al.* [19] présentent de nombreuses manières de faire ce calcul. Les protocoles que nous avons vu jusqu'à présent font généralement une addition des différents témoignages ou une moyenne. Les auteurs précisent que si ces méthodes de calcul permettent une compréhension de tous, elles ne permettent pas une gestion aussi fine de la réputation qu'on le voudrait : un fournisseur de service ayant reçu 90 témoignages positifs et 10 négatifs aura le même score qu'un autre ayant reçu 80 témoignages positifs. De plus, ces méthodes ne permettent pas de donner plus d'importance à des témoignages émanant d'agents particuliers.

D'autres méthodes de calculs de scores de réputation permettent de combler ce manque, comme le modèle discret décrit par Abdul-Rahman et Hailes [3]. Les auteurs proposent de représenter la confiance d'un agent p en un fournisseur de service FS par un degré de confiance discret qui ne peut prendre que quatre valeurs : très digne de confiance, digne de confiance, peu digne de confiance et absolument pas digne de confiance. Pour chaque autre agent, p a également une « confiance de recommandation », lui permettant d'accorder plus d'importance aux témoignages venant de sources sures et de diminuer l'importance des témoignages incertains. Cependant, les auteurs reconnaissent que ce système souffre d'un problème d'initialisation puisqu'un nouvel agent ne sait pas à quels témoins se fier.

Jøsang [17] décrit un modèle « d'opinion ». Son principe est d'avoir un quadruplet (b, d, u, a) exprimant la croyance de p en une déclaration binaire concernant un fournisseur de service FS comme « FS est digne de confiance ». Soit FS est digne de confiance, soit il ne l'est pas. Cependant il n'est pas possible de déterminer avec certitude si cette déclaration est vraie ou fausse. Un agent fait donc preuve d'une *opinion* à propos d'une telle déclaration. Cela se traduit par des degrés de conviction ou de méfiance, un dernier paramètre étant nécessaire pour caractériser l'incertitude. Les paramètres b, d et u repré-

sentent respectivement la conviction (*belief*), la méfiance (*disbelief*) et l'incertitude (*uncertainty*), avec $b, d, u \in [0, 1] \mid b + d + u = 1$. Cette égalité représente le fait que la conviction d'une personne croît en même temps que sa méfiance diminue, c'est-à-dire qu'une personne sûre de l'affirmation d'une déclaration n'est pas méfiante à propos de cette déclaration. Le paramètre u nuance les deux premiers paramètres dans le sens où quelqu'un peut ne pas être sûr ni d'une affirmation, ni de sa négation. Ici, a est un paramètre qui détermine à quel degré l'incertitude contribue au calcul du score de réputation. En ayant une opinion comme « FS est digne de confiance », la réputation de FS sera

$$r = b + u \times a$$

En considérant deux opinions $\omega_1 = (b_1, d_1, u_1, a_1)$ et $\omega_2 = (b_2, d_2, u_2, a_2)$ à propos d'une même déclaration, on peut obtenir leur somme (c'est-à-dire une moyenne des deux opinions) $\bar{\omega} = (\bar{b}, \bar{d}, \bar{u}, \bar{a})$ comme suit :

$$\begin{cases} \bar{b} &= (b_1 u_2 + b_2 u_1)/\kappa \\ \bar{d} &= (d_1 u_2 + d_2 u_1)/\kappa \\ \bar{u} &= (u_1 u_2)/\kappa \\ \bar{a} &= (a_2 u_1 + a_1 u_2 - (a_1 + a_2) u_1 u_2)/(u_1 + u_2 - 2 u_1 u_2) \end{cases}$$

où $\kappa = u_1 + u_2 - u_1 u_2$ et $u_1, u_2 \notin \{0, 1\}$. Lorsqu'un agent p recommande un autre agent FS avec l'opinion $\omega_1 = (b_1, d_1, u_1, a_1)$ et que l'opinion de FS à propos d'une déclaration x est $\omega_2 = (b_2, d_2, u_2, a_2)$, l'opinion de p à propos de x via la recommandation de FS est $\omega_R = (b_R, d_R, u_R, a_R)$ telle que :

$$\begin{cases} b_R &= b_1 b_2 \\ d_R &= b_1 d_2 \\ u_R &= d_1 + u_1 + b_1 u_2 \\ a_R &= a_2 \end{cases}$$

L'auteur nous explique cependant que certaines situations ne peuvent être analysées complètement sans ignorer certains témoignages. En effet, un tel réseau peut être vu comme un graphe orienté où les nœuds sont les agents et les arcs les recommandations. Lorsque deux chemins sont possibles, il faut choisir un unique chemin à prendre en compte, ce qui revient à abandonner certains témoignages. De plus, ce système n'est pas intuitif comme peut l'être une méthode de calcul de réputation additive, et prendre deux témoignages différents en compte est déjà complexe.

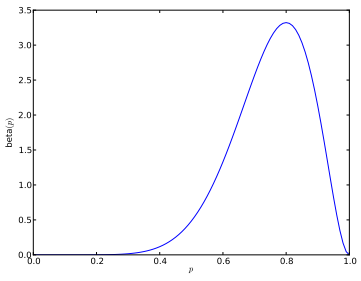
Enfin, Jøsang et Ismail [18] décrivent la beta-réputation, un exemple de système bayésien. Ce système suppose que le comportement d'un agent suit une loi de probabilité beta de paramètres a et b inconnus. En effet, un fournisseur de service réputé peut être vu comme une personne qui se comporte correctement dans la majorité des cas mais à qui il peut arriver des mésaventures. Les différents témoignages collectés permettent d'inférer la distribution de probabilité les ayant généré. Les auteurs présentent un système où les retours sont soit positifs, soit négatifs, et où le score de réputation est mis à jour à chaque itération avec les nouveaux retours. Le score est représenté sous la forme d'une densité de probabilité de loi beta de paramètre (a, b), où a est le

nombre de retours positifs et b celui des retours négatifs :[3]

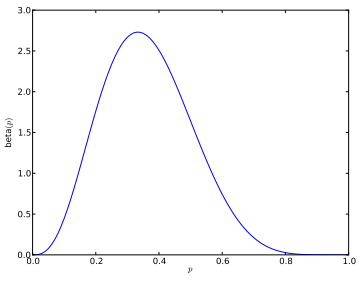
$$\text{beta}(p|a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} p^{a-1}(1-p)^{b-1}, p \in [0,1]$$

Cette fonction représente un score de réputation en fonction de la probabilité qu'un agent l'atteigne. Initialement (c'est-à-dire pour $a = b = 1$), elle est uniforme. À chaque fois qu'un nouveau témoignage est émis, cette fonction (et donc le comportement estimé) est modifiée. L'espérance mathématique de cette fonction vaut $a/(a+b)$, c'est-à-dire le rapport entre le nombre de témoignages positifs et le nombre de témoignages totaux, ce qui concorde avec une moyenne des témoignages tout en apportant plus de précisions. Ainsi, le comportement d'un fournisseur de service est modélisé par une variable aléatoire, de densité de probabilité $\text{beta}(p|a,b)$.

Le comportement étant modélisé par une variable aléatoire continue, la probabilité que le comportement de FS vaille une note spécifique pour une transaction est infiniment faible. Cependant, la probabilité que le comportement vaille une note comprise dans un certain intervalle de $[0,1]$ est non nulle. En effet, $P(p \in [p_1, p_2]) = \int_{p_1}^{p_2} f(p|a,b)$. La Figure 2.2 présente deux scores de réputation : (a) est le score de réputation – c'est-à-dire la densité de probabilité beta – après sept retours positifs et un retour négatif, où on voit que la probabilité que FS se comporte de manière à avoir une note dans l'intervalle $[0.6, 0.9]$ est très grande, tandis que (b) est le score de réputation après deux retours positifs et cinq négatifs, où le comportement de FS a une forte probabilité de valoir une note dans l'intervalle $[0.2, 0.6]$.



(a) Score de réputation pour $a = 8, b = 2$



(b) Score de réputation pour $a = 3, b = 6$

FIG. 2.2: Exemples de scores de réputation positifs et négatifs

L'inconvénient majeur de cette méthode est que les retours sont binaires : soit la transaction s'est bien déroulée, soit elle s'est mal déroulée.

Whitby *et al.* [36] proposent plusieurs améliorations à cette méthode. La première autorise des retours non-binaires. Cela se fait en considérant un témoignage non plus comme un retour positif ou négatif, mais comme un vecteur :

$$\rho = \begin{bmatrix} \rho_+ & \rho_- \end{bmatrix}, \rho_+, \rho_- \in [0,1] \mid \rho_+ + \rho_- = 1$$

[3] Γ est la fonction étendant la factorielle aux nombres complexes.

où ρ_+ représente la partie positive du témoignage et ρ_- sa partie négative. La deuxième amélioration permet de vieillir les témoignages, c'est-à-dire de donner plus d'importance aux transactions récentes qu'aux transactions plus vieilles. Cette étape se fait en introduisant une « fonction de vieillissement » $f(t)$. Si t_ρ représente le temps écoulé depuis l'émission du témoignage ρ, positif – ou nul si le témoignage vient d'être émis –, le nouveau témoignage est :

$$\overline{\rho} = \rho \times f(t_\rho)$$

Pour cette fonction de vieillissement, les auteurs introduisent un facteur de longévité λ et utilisent $f : t_\rho \mapsto \lambda^{t_\rho}$. Si le témoignage $\rho = \begin{bmatrix}\rho_+ & \rho_-\end{bmatrix}$ a été émis à l'instant $t - t\rho$, alors la valeur utilisée dans le calcul du score de réputation est $\overline{\rho} = \begin{bmatrix}\lambda^{t_\rho}\rho_+ & \lambda^{t_\rho}\rho_-\end{bmatrix}$.

En plus de ces améliorations, Whitby *et al.* [36] proposent un algorithme de filtrage des témoignages « injustes » c'est-à-dire trop éloignés du score de réputation d'un fournisseur de service. Pour celui-ci, ils considèrent l'ensemble T_{FS} de tous les témoignages portant sur le fournisseur de service FS, et construisent l'ensemble F_{FS} des témoignages « filtrés » sur FS. Les témoignages qui sont trop éloignés du score de réputation sont ensuite retirés itérativement, jusqu'à ce que F_{FS} soit stable. Cela se fait en calculant le score de réputation (c'est-à-dire la beta-fonction) générée par les témoignages d'un client et en comparant son premier et dernier centile[4] à la moyenne des témoignages. Si le premier centile est supérieur à cette moyenne ou si le dernier centile lui est inférieur, on enlève le témoignage de F_{FS}.

Un avantage de cette méthode est que d'après Fiser [15], chercheur en psychologie cognitive, le calcul bayésien est le modèle qui décrit le mieux notre méthode d'apprentissage ce qui permet de représenter la réputation de manière similaire à ce que nous faisons naturellement. Jøsang *et al.* [19] rappellent que l'inconvénient majeur des systèmes bayésiens est qu'ils sont difficiles à comprendre pour un utilisateur non expert.

2.3 Attaques ciblant les systèmes de réputation

Les systèmes de réputation sont sensibles à des degrés variables aux attaques d'agents malveillants qui tentent par tous les moyens possibles de tirer profit des vulnérabilités de ces systèmes. De nombreux articles s'intéressent à ce sujet [9] [11] [16] [19] [22] [25] [28]. En plus des attaques portant sur les mécanismes des systèmes de réputation, d'autres peuvent exploiter directement le réseau sous-jacent.

2.3.1 Attaques spécifiques aux systèmes de réputation

Carrara et Hogben [9] présentent de nombreuses attaques ciblant les mécanismes inhérents aux systèmes de réputation. Certaines ont une portée globale tandis que d'autres sont plus locales. Pour l'attaque de Sybil [11], un attaquant revendique plusieurs identités dans le système, ce qui lui permet de contrôler multiples nœuds, augmentant ainsi son influence sur le réseau – il peut poster de nombreux témoignages fallacieux. L'utilisation d'une autorité de

[4] Le q-quantile d'une distribution de variable aléatoire X est le plus petit x vérifiant $P(X \leq x) \geq q$

démarrage, demandant un coût avant de permettre l'entrée d'un agent dans le système, diminue le risque des attaques de Sybil. Cela peut se faire avec des puzzles calculatoires comme ceux présentés par Borisov [8]. Plusieurs attaquants peuvent également former une *collusion*. C'est-à-dire qu'ils mettent en commun leurs ressources et connaissances afin d'obtenir encore plus d'informations sur un autre agent ou de modifier la réputation d'un fournisseur de service.

Si jamais la réputation initiale d'un agent est trop élevée, un attaquant peut tromper des clients potentiels, le croyant bienveillant. Au contraire, si la réputation initiale est trop faible, les nouveaux arrivants seront découragés et il sera difficile pour un nouveau fournisseur de service d'attirer des clients. C'est le *problème d'initialisation*.

D'autres attaques sont plus locales, permettant de modifier la réputation d'un fournisseur de service particulier.

Pour l'*attaque par blanchiment de réputation*, un attaquant réinitialise sa réputation lorsqu'il la juge trop faible. Un attaquant peut aussi vouloir *filtrer* l'ensemble des témoignages concernant un fournisseur pour augmenter la proportion de témoignages favorables et ainsi augmenter sa réputation.

Si un attaquant veut modifier la réputation d'un fournisseur de service – que ce soit pour l'améliorer ou la diminuer –, il peut également faire du *bourrage d'urne*, c'est-à-dire émettre de nombreux témoignages fallacieux pour donner l'impression que le fournisseur est soit bienveillant, soit malveillant.

Des attaques visant un algorithme ou une architecture sont également exploitées. Un exemple très connu est celui des « Google bomb » [2]. Cette technique exploitait l'algorithme PageRank utilisé par le moteur de recherche Google, donnant un score au texte source contenant un hyperlien vers une autre page. Plus nombreux sont les sites utilisant un même texte source, plus élevé sera ce score. À partir d'un certain score, la page vers laquelle pointe le lien apparaît dans les résultats lors d'une recherche du texte source, même si ce texte n'apparaît pas dans la page obtenue. En pratique, un attaquant peut utiliser une multitude de sites web et les faire pointer vers un même site (par exemple `http://www.example.com`) en utilisant un même texte source (par exemple « ce site est magnifique »). En cherchant « site magnifique » dans Google, le site `http://www.example.com` apparaîtra dans les résultats même s'il ne contient ni le mot « site », ni le mot « magnifique ». Ce genre de techniques est appelé « attaques contre le score de réputation par manipulation des critères d'évaluation ».

Un attaquant peut vouloir *médire* sur un fournisseur de service en apportant des témoignages de mauvaise qualité. Ces attaques par *médisance* peuvent être amplifiées si l'attaquant peut se créer de nombreuses identitées, par exemple grâce à une attaque de Sybil réussie ou via une collusion. Finalement, un attaquant peut également essayer de *réfuter* une transaction où le fournisseur de service s'est bien comporté pour éviter d'avoir à émettre un témoignage positif.

2.3.2 Attaques génériques contre les systèmes distribués

Comme nous l'avons vu, dans une architecture centralisée, la sécurité du système repose sur la sécurité du serveur central. Si jamais celui-ci est compro-

mis, le fonctionnement du système le sera aussi. C'est donc un point unique de défaillance. C'est ce fait qui nous motive à considérer un système distribué. Cependant, un système distribué est plus compliqué à mettre en place qu'un système centralisé du fait de l'absence de serveur central. Urdaneta *et al.* [35] présente les principales attaques possibles sur les réseaux sous-jacents, au nombre de trois : attaque de Sybil, attaque d'éclipse et attaque sur le routage :

L'attaque de Sybil, dont nous avons parlé précédemment, permet à un attaquant de contrôler plusieurs nœuds du réseau. Dans un réseau, cela lui permet par exemple de router un paquet où bon lui semble.

L'attaque d'éclipse corrompt la table de routage d'un nœud honnête pour s'assurer que les paquets passeront par un nœud contrôlé. Le nœud honnête est alors « éclipsé » puisque la plupart de ses communications peuvent être modifiées par un autre nœud.

Les attaques de routage consistent principalement à ne pas retransmettre ou à modifier les requêtes. Leur impact est augmenté si elles sont combinées avec des attaques de Sybil et d'éclipse.

La plupart des solutions aux attaques d'éclipse et de routage ou de stockage présentées [28] [30] [31] [35] font intervenir des chemins multiples, afin de réduire les problèmes concernant le réseau sous-jacent. En effet, Chord est déjà robuste de par sa construction, et sa robustesse peut être améliorée pour tolérer plus d'attaques comme le montrent Artigas *et al.* [6] et Fiat *et al.* [14]. Augmenter le nombre de chemins augmente cette robustesse.

Un autre problème qui se pose lors de communications dans un réseau distribué est celui de la gestion des identités. En effet, il est parfois nécessaire d'être sûr de l'identité de l'agent avec lequel il communique. Marti et Garcia-Molina [22] proposent à cet effet des techniques de clé publique/privée pour éviter le vol d'identité. Néanmons, ce système n'est fiable que s'il existe une tierce partie de confiance, un serveur centralisé faisant office de CA (*Certificate Authority*) et générant des certificats sur les clés publiques.

2.4 Système de réputation centralisé préservant la vie privée

Il y a déjà eu des tentatives de systèmes de réputation centralisés protégeant la vie privée. À notre connaissance, le seul système *prouvé* préservant l'anonymat de ses utilisateurs est le système proposé par Androulaki *et al.* [5]. L'objectif de ce système est de fournir une architecture de commerce électronique préservant la vie privée. L'architecture repose sur la présence d'une banque, agent central chargé de lier les identités des agents et leur réputation. La réputation d'un agent est matérialisée par des jetons, qui peuvent être regroupés dans des portefeuilles pour constituer la monnaie d'échange. Ils sont appelés *repcoin*, pour *Reputation Coin*. Plus un agent possède de repcoin, meilleure est sa réputation. Ils jouent donc un double rôle, à la fois de monnaie et de points de réputation. Afin de garantir leur anonymat, les agents discutent entre eux via des pseudonymes, tandis qu'ils utilisent leur identité réelle pour communiquer avec la banque.

Ce système repose sur un réseau de communication anonyme, par exemple un routage en oignon comme celui présenté par Syverson *et al.* [34]. Le principe de ce type de routage est d'éviter toute traçabilité entre la source

d'un message et son destinaire. Ainsi, au lieu de se connecter directement à un utilisateur distant, un utilisateur du routage en oignon utilise plusieurs intermédiaires successifs afin d'éviter que quiconque puisse détecter l'interaction entre les deux utilisateurs. Les auteurs supposent également que chaque agent possède un nombre limité de repcoin et ne peuvent donc en donner qu'une quantité finie par unité de temps. Cette limite peut être représentée par une cotisation journalière ou hebdomadaire nécessaire pour participer au système. De plus, n'importe quel nombre d'agents peut conspirer (y compris la banque) pour révéler l'identité d'un autre agent.

La banque gère toutes ces informations en utilisant des paires de clés concernant les identités. Elle utilise également trois bases de données, la première concernant les quotas de repcoin (utilisés pour incrémenter automatiquement le solde), la seconde gérant les scores de réputation, c'est-à-dire le solde de repcoin, tandis que la dernière conserve un historique des transactions.

Ce système de réputation assure les propriétés présentées ci-dessous.

Conformité Lorsqu'un agent honnête U_1 demande à retirer des repcoin à une banque honnête B et qu'il en a assez, il n'y aura pas de message d'erreur ; s'il donne une repcoin issue de ce retrait à un autre agent honnête U_2, U_2 l'acceptera ; si U_2 dépose cette repcoin dans une banque honnête, alors sa réputation sera incrémentée. Quand un agent honnête demande une confirmation d'identité (pour laquelle il est éligible) à une banque honnête, il aura une identité valide.

Impossibilité de faire une liaison identité - pseudonyme Pour un attaquant qui peut avoir corrompu certaines parties (y compris la banque), il n'y a pas d'autre moyen que de supposer aléatoirement qu'un pseudonyme et une identité (ou deux pseudonymes) sont liés.

Pas de sur-attribution Aucun ensemble d'agent ne peut attribuer plus de repcoin qu'ils n'en ont retiré.

Disculpation Aucune coalition d'agents ne peut forger une preuve qu'un agent honnête a dépensé deux fois la même repcoin.

Inforgeabilité de la réputation Aucune coalition d'agents de réputation au plus l ne peut montrer une réputation plus grande que l pour un de ses pseudonymes.

La figure 2.3 présente le protocole d'attribution de protocole. À gauche, l'utilisateur U donne des repcoin à M, et à droite M prouve à U qu'il a assez de réputation.

Pour retirer et attribuer des repcoin, le protocole est le suivant :

1. l'utilisateur U retire un portefeuille W de la banque B comprenant un certain nombre de repcoin ;
2. U donne une repcoin (S, π) de son portefeuille à M ; l'échange se fait via les pseudonymes P_U et P_M ;
3. P_M dépose la repcoin (S, π) à la banque ;
4. P_M obtient une permission aveugle σ de la banque, visible uniquement par M ;
5. finalement, M dépose σ à la banque, et B augmente le score de réputation de M.

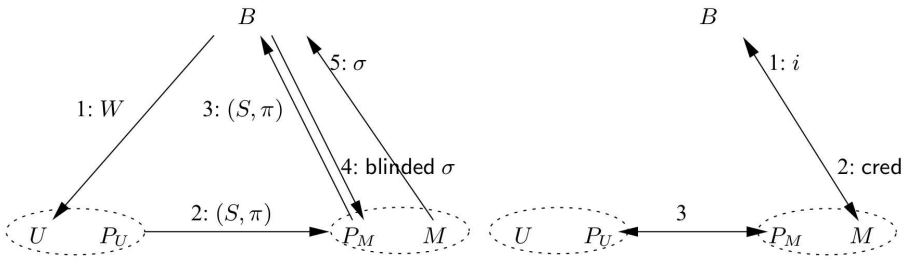


FIG. 2.3: Exemple d'attribution de réputation et de preuve de réputation [5]

Lorsque M veut prouver à U que sa réputation a atteint un certain niveau, le protocole suivant est exécuté :

1. M demande une identité pour le groupe G_i ;
2. si la réputation de M est assez élevée, B génère une identité cred à M ;
3. P_M utilise cred pour prouver à P_U qu'il appartient à G_i.

Ce système permet d'assurer l'anonymat de ses utilisateurs dans le cas particulier de micro-transactions électroniques en présence d'une tierce partie de confiance (la banque), à travers l'utilisation de multiples identités : la banque connaît uniquement les identités réelles tandis que les autres utilisateurs connaissent uniquement des pseudonymes générés aléatoirement.

Les auteurs précisent que le défaut de ce système est l'absence de témoignages négatifs : un utilisateur peut seulement savoir qu'un fournisseur de service s'est comporté correctement m fois, ce qui est loin de fournir un indicateur aussi précis que savoir qu'il s'est bien comporté sur telle proportion de ses transactions. De plus, la banque est un point unique de défaillance du système.

Dans la suite, nous proposons une architecture de système de réputation distribuée permettant de respecter la vie privée des clients. À notre connaissance, il n'existe aucun système de réputation distribué préservant l'anonymat des agents.

Chapitre 3

Objectifs

Avant de présenter les objectifs recherchés avec notre proposition, nous revisitons la terminologie présentée au chapitre 2 pour introduire des notions propres à notre solution. Nous complétons également le modèle de l'adversaire. Nous présentons ensuite les propriétés assurées en expliquant pourquoi elles sont importantes.

3.1 Terminologie et définitions

Un *témoignage* est un ensemble de données permettant d'évaluer une transaction. Il comporte l'opinion d'un client à propos d'un fournisseur de service, fondé sur ses interactions passées avec lui et l'opinion du fournisseur de service sur les transactions auxquelles il a participé avec ce même client.

Afin de garantir le respect de leur vie privée, les clients agissent sous le couvert de pseudonymes. Les témoignages sont stockés sur des agents bien identifiés fournissant un service de *boîte aux lettres*. Cette notion sera précisée au Chapitre 4.

Définition 1 (Contenu d'un témoignage)**.** *Afin d'obtenir un score de réputation pertinent, le témoignage d'un agent utilisant un pseudonyme p sur un fournisseur de service FS doit porter plusieurs informations :*

- *la* note *donnée par p sur le comportement de FS ;*
- *la* note *donnée par FS sur le déroulement de la transaction ;*
- *une estampille pour rendre compte de la date à laquelle la transaction a lieu ;*
- *la valeur de la transaction pour valoriser les transactions importantes.*

On note ω un témoignage. Les différents témoignages concernant un fournisseur de service FS sont agrégés par chaque nouveau client désirant se forger une opinion sur FS. Plus précisément, ces témoignages sont agrégés pour calculer un *score de réputation* dont la pertinence dépend à la fois de la quantité des témoignages collectés et de leur qualité. Un témoignage est dit *de qualité* si chacun des deux avis portés reflète les comportements des deux agents concernés.

Le comportement d'un agent (client ou fournisseur de service) se divise en deux parties : la conformité et l'honnêteté.

Définition 2 (Conformité)**.** *Un agent est dit* correct *s'il suit le protocole du système de réputation tout au long de ses interactions ; il est dit* incorrect *sinon.*

C'est une donnée objective et binaire : soit le protocole est suivi – et la note vaut alors 1 –, soit il n'est pas suivi – et la note vaut alors 0. Un agent quittant le réseau au milieu d'une interaction est incorrect.

Définition 3 (Honnêteté)**.** *Un client est dit* honnête *si chaque témoignage qu'il émet reflète son jugement du comportement du fournisseur de service concerné. Un fournisseur de service est dit* honnête *si son comportement pour chaque transaction est tel qu'il l'aurait jugé bon si lui même avait été son propre client. Dans le cas contraire, un agent est dit* malhonnête*.*

L'honnêteté est une donnée subjective dont l'interprétation se rapproche de la notion juridique de *bonus pater familias* – « bon père de famille » –, qui représente la norme comportementale d'un individu mais n'est pas formellement définie. L'honnêteté est notée dans l'intervalle $[0, 1]$. Plus globalement, on définit la *bienveillance* comme suit :

Définition 4 (Bienveillance)**.** *Un agent est dit* bienveillant *s'il est à la fois honnête et correct. Dans le cas contraire, il est dit* malveillant*.*

Nous considérons ici que les attaquants sont des agents du système, c'est-à-dire qu'ils fournissent ou utilisent un service. Dans un premier temps, nous supposons que seuls les clients peuvent lancer des attaques. Dans un second temps, nous élargirons les attaquants à tous les agents du système.

De plus, nous nous focalisons sur le système de réputation, autrement dit la partie applicative du système, c'est pourquoi nous supposons que le réseau est fiable, c'est-à-dire que les attaques de Sybil, d'éclipse ou de routage et de stockage présentées à la Section 2.3 sont tolérées par des mécanismes adéquats. Comme nous l'avons précisé au chapitre 2.3, de tels mécanismes existent.

Hypothèse 1 (Robustesse du réseau)**.** *Les opérations de routage sont fiables, à savoir toute requête émise par un agent source est reçue par son destinataire en un temps borné et connu.*

Afin de définir les propriétés de vie privée qui nous intéressent, nous nous inspirons du travail de formalisation de Pfitzmann et Hansel [27]. Ainsi, nous définissons *pseudonymat* et *non-traçabilité* de la manière suivante :

Définition 5 (Pseudonymat)**.** *Une entité est dite* pseudonyme *si elle est connue uniquement via des pseudonymes, c'est-à-dire des identifiants différents de sa réelle identité.*

Définition 6 (Non-traçabilité)**.** *Deux entités sont dites* non-traçables *si un attaquant n'est pas capable de dire si elles représentent la même entité ou non.*

3.2 Objectifs

L'objectif de ce stage est de concevoir un système de réputation *robuste* et *préservant la vie privée* de ses agents en présence d'attaques de ces agents. Un tel système vérifie les propriétés suivantes :

Propriété 1 (Robustesse)**.** *Au bout d'un temps fini, le score d'un agent A calculé par un agent bienveillant reflète le comportement de A.*

Propriété 2 (Respect de la vie privée)**.** *On dit que le système préserve la vie privée de ses clients si les trois conditions suivantes sont satisfaites :*

- *les clients sont pseudonymes ;*
- *une identité et un pseudonyme ne sont pas traçables ;*
- *deux pseudonymes ne sont pas traçables.*

Si la première propriété est vérifiée, le système de réputation est dit *robuste*. Si la dernière est également vérifiée, on dira qu'il *préserve la vie privée*.

Chapitre 4

Proposition

Dans ce chapitre, nous présentons notre proposition. Dans un premier temps, nous exposons un système de réputation distribué et préservant la vie privée de ses clients en supposant qu'un fournisseur de service ne cherche pas à modifier les témoignages le concernant. Cette hypothèse étant irréaliste, nous raffinons notre proposition dans la Section 4.2. Finalement, nous proposons un protocole d'interaction dans la Section 4.3

4.1 Système de réputation en présence de fournisseurs honnêtes

Dans un premier temps, nous présentons la manière dont est garanti le pseudonymat des clients. Ensuite, nous exposons l'organisation du réseau. Après cela, nous présentons le calcul du score de réputation. Finalement, nous proposons une modélisation du système de réputation.

4.1.1 Pseudonymat des clients

Afin de garantir le pseudonymat des clients, ceux-ci ne communiquent jamais sous leur identité réelle dans le réseau, mais utilisent plutôt des pseudonymes. Les clients génèrent eux-mêmes leurs pseudonymes qu'ils font ensuite certifier par une autorité de démarrage (AD). Afin de limiter le nombre de pseudonymes par client, chaque certification possède un coût. Le système de monnaie électronique Bitcoin [24], permettant de payer anonymement tout en conservant les caractéristiques d'une monnaie « classique », peut être utilisé à cet effet.

Lorsqu'un client A veut utiliser les services du réseau, il commence par générer autant de pseudonymes qu'il estime nécessaire et un couple de clés publique/privée pour chacun d'entre eux. Le i-ième pseudonyme de A est noté $p(A, i)$ ou plus simplement p lorsque la situation ne prête pas à confusion. Ensuite, A communique avec l'autorité de démarrage et la paye pour qu'elle certifie ses pseudonymes. Une fois que ses pseudonymes sont certifiés, A peut prendre place dans le réseau suivant la valeur de ses pseudonymes. Par exemple, si une DHT est utilisée, les positions de A dans celle-ci seront fonction des valeurs des pseudonymes $p(A, 1), \ldots, p(A, n_A)$. À tout instant, A peut contacter à nouveau l'autorité de démarrage pour obtenir plus de pseudonymes.

4.1.2 Organisation du réseau

Le système de réputation proposé repose sur Chord [32], une DHT dans laquelle sont placés les fournisseurs de service. Cela permet aux clients de les localiser et de communiquer avec eux efficacement, c'est-à-dire en nombre de sauts poly-logarithmique en la taille du système. Comme les fournisseurs de service ne sont pas supposés chercher à modifier les témoignages, ils peuvent conserver eux-même les témoignages les concernant. Cette organisation est présentée à la Figure 4.1.

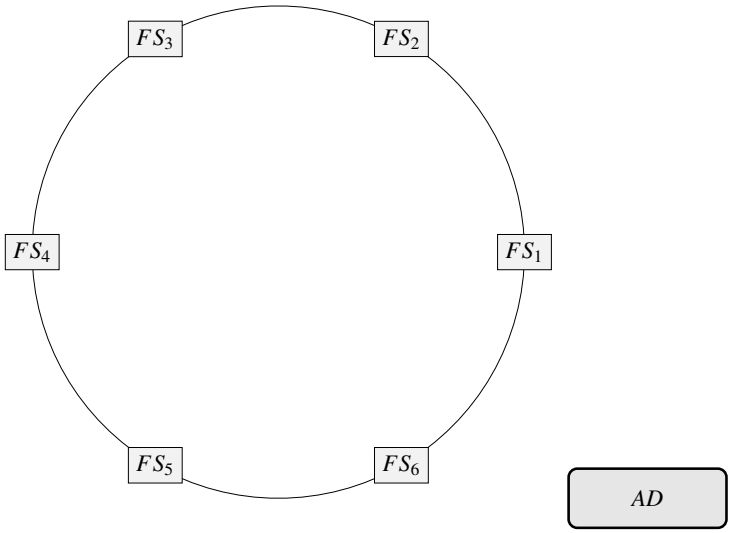


FIG. 4.1: Organisation naïve du réseau

Afin d'empêcher les attaques visant à augmenter artificiellement le nombre de témoignages (*bourrage d'urnes*), dès qu'un fournisseur de service reçoit un nouveau témoignage d'un client avec qui il a déjà interagi, l'ancien témoignage est remplacé. Ainsi, à tout instant, et pour chaque paire d'agents ayant interagi, il n'y a qu'une seule paire de témoignages entre ces agents. Cela modifie le concept de réputation par rapport aux systèmes de réputation classiques. En effet, si jamais deux agents interagissent beaucoup – par exemple un restaurateur et un de ses fournisseurs –, dans un système usuel il y aurait autant de témoignage que d'interactions. Dans le système proposé, il n'y a qu'un seul témoignage représentant l'opinion générale du restaurateur sur son fournisseur.

4.1.3 Calcul des scores de réputation

La méthode bayésienne décrite par Jøsang et Ismail [18] et améliorée par Whitby *et al.* [36] est utilisée pour calculer la réputation d'un fournisseur, vieillir les témoignages et filtrer les témoignages injustes.

Le principal problème est que cette fonction de calcul de score de réputation ne prend en compte que les notes unilatérales, c'est-à-dire les notes des clients d'un fournisseur de service à propos de celui-ci. L'avis du fournisseur de service sur la transaction n'a aucun impact sur le score de réputation, ce qui

peut être préjudiciable à un fournisseur de service lorsqu'il interagit avec des clients malveillants. Une manière naïve de prendre en compte l'avis du fournisseur est de lui demander de noter l'interaction, puis de faire la moyenne des deux notes. Cependant, cette façon de procéder incite un fournisseur de service malveillant à surnoter le déroulement de la transaction. Ainsi les témoignages le concernant seront toujours positifs au moins à moitié. Afin d'inciter les participants à donner des témoignages de qualité – c'est-à-dire véridiques –, la différence entre la note donnée par le client et celle donnée par le fournisseur de service est utilisée afin de *moduler* la moyenne des deux. Cela se fait via une fonction de corrélation f_{corr}, appliquée sur la différence entre les deux notes : si la différence est faible, le témoignage résultant sera meilleur que la moyenne des deux. Au contraire, si cette différence est élevée, le témoignage résultant sera moins bon que la moyenne des deux témoignages. En notant ρ_{FS}^{p} la note émise par le fournisseur de service et ρ_{p}^{FS} celle du pseudonyme, toutes deux concernant la même transaction et $\widetilde{\rho}$ le témoignage résultant, cela se traduit de la manière suivante :

$$\widetilde{\rho} = \frac{\rho_{FS}^{p} + \rho_{p}^{FS}}{2} + f_{\text{corr}}\left(\left|\rho_{FS}^{p} - \rho_{p}^{FS}\right|\right)$$

Nous désignons par *modulation* cette étape du calcul du score d'un fournisseur de service. Nous proposons d'utiliser la fonction suivante comme fonction de corrélation :

$$f_{\text{corr}} \ : \ x \in [0,1] \mapsto \begin{cases} \frac{m_+}{l}\,(l-x) \ \text{ si } x < l \\ \frac{m_-}{1-l}\,(x-l) \ \text{ sinon} \end{cases} \qquad \text{avec } 0 \leq l \leq 1$$

C'est la fonction affine qui ajoute m_+ si les deux notes sont identiques, 0 si la différence vaut l, et qui retire $m_- < 0$ si l'écart est maximum (c'est-à-dire s'il vaut 1). La figure 4.2 présente la note modulée en fonction de la note donnée par le fournisseur de service, pour différentes notes données par le pseudonyme pour $m_+ = 0.05, l = 0.1$ et $m_- = -0.6$. Cette figure montre que pour maximiser son score de réputation, un fournisseur de service a intérêt à donner une note réaliste pour toutes ses interactions. En effet, si jamais il surnote le déroulement de la transaction et qu'il donne quand même une bonne note, il obtiendra une mauvaise note. En revanche, s'il se comporte bien et qu'un client médit, il héritera quand même d'une mauvaise note. Cela laisse l'avantage au client. Cependant, le filtrage des témoignages de [36] permet de détecter les témoignages fallacieux et de les écarter.

Une fois que chaque témoignage a été modulé, la valeur de la transaction concernée doit également être prise en compte. Cela se fait tout simplement en multipliant les notes obtenues par la valeur v de la transaction :

$$\widehat{\rho} = \widetilde{\rho} \times v$$

Ensuite, les témoignages doivent être veillis afin de donner plus d'importance aux transactions récentes, comme proposé dans [36]. Cela se fait avec une « fonction de vieillissement » $f(t) = \lambda^t$. Si t_ρ représente le temps écoulé depuis l'émission du témoignage ρ, positif – ou nul si le témoignage vient d'être émis –, le nouveau témoignage est :

$$\overline{\rho} = \widehat{\rho} \times f(t_\rho)$$

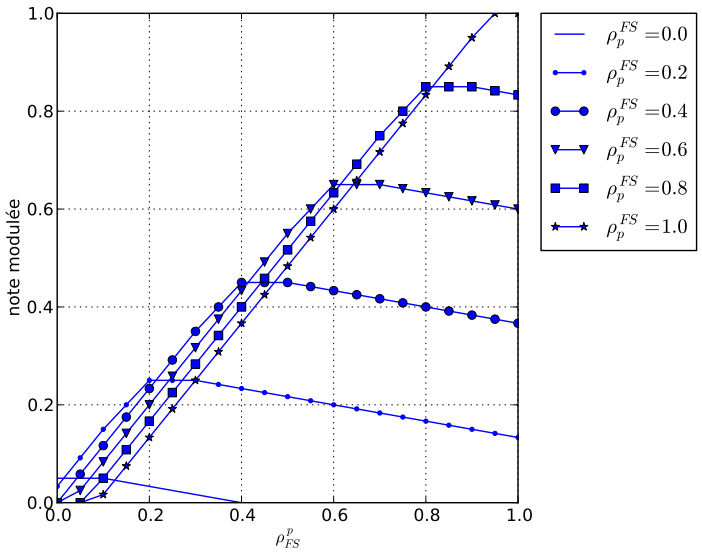


FIG. 4.2: Note modulée en fonction de la note du fournisseur de service $\rho_F S^p$, pour plusieurs notes du client ρ_p^{FS}

Afin d'écarter les témoignages des clients médisants, la méthode de filtrage présentée par Whitby *et al.* [36] est utilisée. L'algorithme utilisé est l'Algorithme 1. Celui-ci présente une différence avec l'algorithme de [36] : en effet, les auteurs conglomèrent tous les témoignages d'un même client avant de calculer leurs quantiles. Parce que l'identité des clients est inconnue et qu'il y a au plus un témoignage par pseudonyme, regrouper les témoignages est impossible. Ainsi, le filtrage doit se faire témoignage par témoignage. Cependant, en considérant un seul témoignage, la beta-réputation a une variance relativement élevée et les quantiles sont trop faibles (resp. trop élevés) pour permettre une détection efficace. En effet, pour un témoignage complètement négatif, les quantiles valent respectivement $q_1 = 0.0$ et $q_{99} = 0.9$. Pour qu'un tel témoignage soit écarté, il faut que la moyenne de la réputation du fournisseur soit supérieure à 0.9. Afin de contourner ce problème, nous accentuons la beta-fonction en multipliant le témoignage par un certain facteur f_F, ce qui augmente le premier centile et diminue le dernier. L'augmentation de ce facteur f_F réduit les faux-négatifs (c'est-à-dire les témoignages fallacieux non filtrés) tout en augmentant les faux-positifs (c'est-à-dire les témoignages authentiques filtrés).

Les différentes étapes du calcul du score d'un fournisseur sont récapitulées à la figure 4.3.

4.1.4 Modélisation du système

Afin de simuler le système, un environnement présentant un fournisseur de service avec qui des pseudonymes interagissent régulièrement est modélisé. Dans un premier temps, les clients sont supposés être honnêtes et cor-

$F_{FS} \leftarrow T_{FS}$
repeat
  $\rho_F = \sum_{\overline{\rho} \in F_{FS}} \overline{\rho}$
  $m = E(\rho_F)$
  for all $\overline{\rho} \in F_{FS}$ **do**
    $s = \text{beta}(1 + f_F \times \overline{\rho}_+, 1 + f_F \times \overline{\rho}_-)$
    $l = q$ quantile de s
    $u = (1 - q)$ quantile de s
    if $l > m$ ou $u < m$ **then**
      $F_{FS} = F_{FS} \setminus \{\overline{\rho}\}$
    end if
  end for
until F_{FS} ne change pas

Algorithme 1: Algorithme de filtrage des témoignages

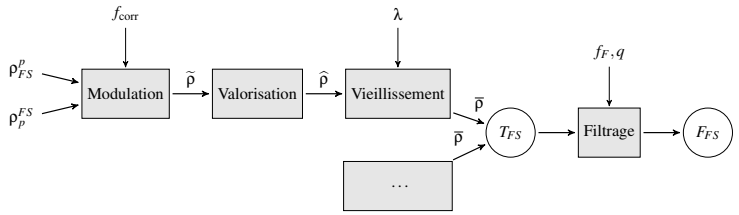


FIG. 4.3: Étapes du calcul du score d'un fournisseur de service

rects, tandis que le fournisseur de service est parfois honnête et/ou correct, parfois malhonnête et/ou incorrect – il ne peut néanmoins pas modifier les témoignages. Le système est paramétré par les variables suivantes :

- le profil r du fournisseur de service, reflétant son comportement ;
- la portée des valeurs des transactions V ;
- le nombre de transactions n_t ;
- le facteur de longévité λ ;
- le temps moyen entre deux transactions t_m ;
- le facteur de filtrage f_F ;
- la proportion de clients malhonnêtes c_c.

Un fournisseur de service FS est modélisé par un automate probabiliste [26] à 5 états : S_i est l'état initial, dans lequel FS attend un nouveau témoignage. À chaque nouvelle transaction, FS se comporte correctement avec une probabilité P_c, et entre alors dans l'état S_1. Sinon, il entre dans l'état S_2. Ensuite, il se comporte honnêtement avec une probabilité P_h et entre alors dans l'état S_3. Sinon, il entre dans l'état S_4. Finalement, il revient à l'état S_i[1] et attend une nouvelle transaction. Lorsque n_t transactions ont été effectuées, FS s'arrête. La Figure 4.4 présente cet automate.

Le profil d'un fournisseur de service est paramétré par 4 éléments : la probabilité de suivre le protocole P_c, la probabilité d'avoir un comportement honnête P_h, l'intervalle dans lequel se trouve la note reçue lors d'un comporte-

[1] ε indique une transition instantanée.

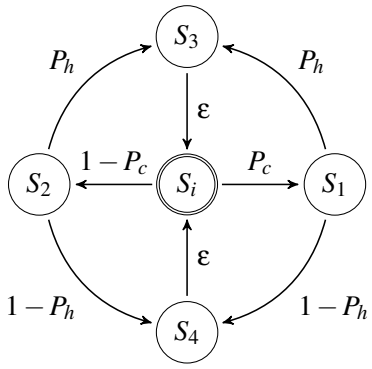


FIG. 4.4: Automate probabiliste modélisant un fournisseur de service

ment honnête I_h et sa contrepartie $I_{\neg h}$ pour un comportement malhonnête.

Les différents profils d'un fournisseur de service sont paramétrés de la façon suivante :

- $r = 1$: FS est honnête et correct dans la plupart des cas : $P_h = 0.8$, $I_h = [0.7, 1]$, $I_{\neg h} = [0.2, 0.7]$ et $P_c = 0.8$;
- $r = 2$: FS est malhonnête et correct dans la plupart des cas : $P_h = 0.2$, $I_h = [0.3, 0.8]$, $I_{\neg h} = [0, 0.3]$ et $P_c = 0.8$;
- $r = 3$: FS est à moitié honnête et à moitié correct : $P_h = 0.5$, $I_h = [0.5, 1]$, $I_{\neg h} = [0, 0.5]$ et $P_c = 0.5$.
- $r = 4$: FS est ce qu'on appelle un « agent dormant ». Le modèle est alors différent : FS se comporte correctement et honnêtement pour n_t transactions de faible valeur avec les mêmes paramètres P_c, P_h, I_h et $I_{\neg h}$ que pour un fournisseur honnête et correct dans la plupart des cas, présentés précedemment. Après ces n_t transactions, il effectuera une dernière transaction de grande valeur pour laquelle il est incorrect et malhonnête avec une probabilité $P_c = P_h = 0$, et reçoit la note 0 pour l'honnêteté et la conformité.

Le temps entre deux témoignages est modélisé par une loi exponentielle de moyenne t_m, ce qui paraît approprié : en effet, la modélisation du temps écoulé entre deux évènements fait souvent appel à cette loi.

4.1.5 Expérimentations

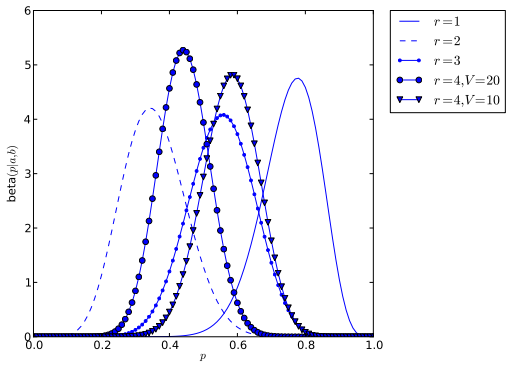
Influence des paramètres du fournisseur de service

Le score de réputation du fournisseur de service après un certain nombre de transactions est étudié en faisant varier les différents paramètres un à la fois. Quand le paramètre ne varie pas ou qu'une mention ne précise pas sa

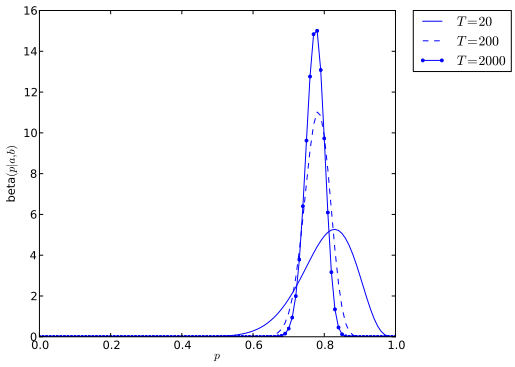
valeur, il est fixé à une valeur par défaut :

$r = 1$	(profil de FS)
$V = [0.5, 2]$	(portée des témoignages)
$n_t = 20$	(nombre de témoignages)
$\lambda = 0.9$	(facteur de longévité)
$t_m = 0.1$	(temps moyen entre deux transactions)
$f_F = 5$	(facteur de filtrage)
$c_c = 0\%$	(proportion de clients malhonnêtes)

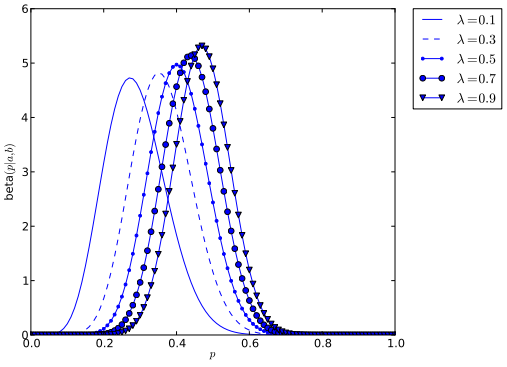
La Figure 4.5 présente ce score de réputation en faisant varier le profil, le nombre de témoignages ainsi que le facteur de longévité.



(a) Score de réputation sur l'honnêteté pour différents profils



(b) Score de réputation sur l'honnêteté pour différents nombres de témoignages



(c) Score de réputation d'un agent dormant ($r = 4$) pour différents facteurs de longévité

FIG. 4.5: Scores de réputation d'un fournisseur de service en faisant varier le profil du fournisseur et le facteur de longévité

Profil du fournisseur de service La Figure 4.5a présente le score de réputation sur l'honnêté de FS en fonction de son profil. Celle-ci montre que le score reflète correctement le profil du fournisseur de service : le score de réputation du profil 1 montre un fournisseur de service se comportant honnêtement dans près de 80% des transactions, celui du profil 2 est honnête dans seulement 30% des transactions, celui du profil 3 est honnête dans environ la moitié des transactions. Notons que pour le profil 3, la variance est élevée, ce qui est normal puisque les deux paramètres de la loi beta sont presque égaux. Finalement le score de réputation d'un agent dormant est moins bon que celui d'un agent honnête. En effet, il s'est comporté malhonnêtement pour la transaction la plus récente, qui avait une grande valeur (10 ou 20), et FS s'y est mal comporté. On constate que l'augmentation de la valeur diminue le score de réputation de FS. Ici, le facteur de longévité est assez important ($\lambda = 0.9$), ce qui explique que la réputation du fournisseur ne soit pas si mauvaise que ça.

Valeurs des transactions et nombre des témoignages L'augmentation des valeurs des transactions réduit l'écart-type des scores tout en modifiant très peu leur moyenne : en effet, cette augmentation accroît les paramètres de la beta-fonction de réputation, augmentant en même temps le coefficient binomial et les puissances. C'est la même chose avec l'augmentation du nombre de témoignages, visible à la Figure 4.5b. Cependant, ceux-ci ne peuvent pas être augmentés à l'infini : en effet, le vieillissement des témoignages diminue peu à peu l'importance des témoignages les plus vieux.

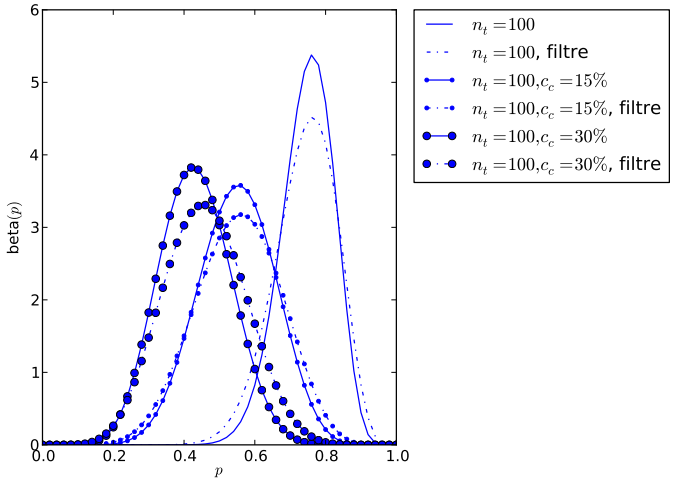
Facteur de longévité et temps moyen entre témoignages Les deux dernières variations concernent le facteur de longévité ainsi que le temps moyen entre deux témoignages et sont liées : en effet, augmenter le temps moyen entre deux témoignages revient à diminuer le facteur de longévité et diminue de plus l'importance des témoignages les plus vieux. L'impact du facteur de longévité est présenté à la Figure 4.5c, où le fournisseur de service se comporte comme un agent dormant ($r = 4$). En diminuant le facteur de longévité – c'est-à-dire en accordant encore moins d'importance aux témoignages anciens –, la variance des deux scores est augmentée. C'est normal : en effet, diminuer le facteur de longévité revient à diminuer l'importance des anciens témoignages. Les paramètres de la beta-fonction sont ainsi réduits, augmentant de cette façon la variance du résultat. Cependant, pour un agent dormant dont la dernière action – malhonnête et incorrecte – vient d'être effectuée, l'accent est mis sur celle-ci, ce qui diminue la moyenne de son score de réputation.

Influence des paramètres du filtrage

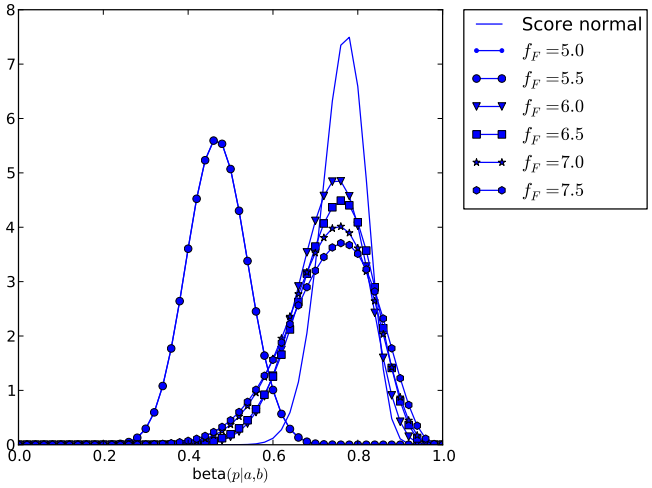
Jusqu'à présent, nous nous sommes intéressés principalement au fournisseur de service et nous avons regardé l'évolution de son score en fonction de quelques paramètres. Nous allons maintenant regarder ce qui se passe en considérant que le fournisseur de service est bienveillant (profil 1) tandis que certains clients sont malhonnêtes et lui affectent une mauvaise note quel que soit le déroulement de la transaction dans l'objectif de diminuer le score du fournisseur de service afin de voir si le filtrage proposé à la Section 4.1.3 est efficace. Ce déroulement est symétrique à celui où le fournisseur de service

est malveillant et où il essaye d'augmenter son score de réputation avec des identités multiples.

Le résultat de l'exécution de l'Algorithme 1 en considérant plusieurs répartitions de clients malveillants pour plusieurs nombres de témoignages est présenté à la Figure 4.6a, pour un facteur de filtrage que l'on fixe naïvement à $f_F = 5$, tandis que le score de réputation en fonction de la valeur du facteur de filtrage est présenté à la Figure 4.6b.



(a) Score de réputation en présence de clients malveillants avec et sans filtrage



(b) Score de réputation en fonction de la valeur du facteur f

FIG. 4.6: Score de réputation d'un fournisseur de service en faisant varier le nombre de clients malveillants et le facteur de filtrage

La Figure 4.6b montre qu'en augmentant la proportion de clients malhonnêtes c_c, le score diminue énormément : pour $c_c = 30\%$, le score de réputation sur l'honnêteté du fournisseur de service est dénote d'un comportement malhonnête alors qu'il se comporte honnêtement. Sans filtrage aucun des témoignages, la fonction de calcul de score de réputation n'est pas robuste aux clients malveillants, et le filtrage ne permet pas une réelle amélioration du score.

Afin d'augmenter la robustesse du calcul du score, nous nous intéressons à l'influence du paramètre f_F sur le score de réputation. Celle-ci est présentée à la Figure 4.6b, pour $c_c = 30\%$, c'est-à-dire un taux relativement élevé de clients malhonnêtes. Afin de pouvoir comparer, cette figure présente aussi le comportement du fournisseur avec la courbe « score normal ». Dans cette instance d'interaction, aucun témoignage n'était filtré pour $f_F \in \{5.0, 5.5\}$ – c'est pourquoi les deux courbes sont superposées –, 33 pour $f_F = 6.0$, 39 pour $f_F = 6.5$, 45 pour $f_F = 7.0$ et 50 pour $f_F = 7.5$. Comme 30 clients étaient malveillants, il y a des faux-positifs dès que $f_F \geq 6.0$. Pour 30% de témoignages malicieux, un facteur de filtrage de 5 était trop faible. Cela montre qu'il y a un palier pour ce facteur de filtrage permettant de filtrer la plupart des témoignages malveillants – voire tous. Avant ce palier, les témoignages ne sont pas filtrés, et au-delà de celui-ci, il y aura de plus en plus de faux-positifs. Dans ce contexte, $f_F = 6$ paraît être une bonne idée pour filtrer efficacement sans avoir trop de faux-positifs. Dans un contexte différent, une valeur différente de f_F peut être plus appropriée pour permettre un filtrage efficace des témoignages. Cette valeur peut être estimée empiriquement de la même manière.

4.2 Système de réputation en présence de fournisseurs malhonnêtes

Nous venons de discuter d'une architecture adaptée à un environnement idéal, dans lequel nous avons fait l'hypothèse que les fournisseurs de service ne modifiaient pas les témoignages les concernant. Il est clair que cette hypothèse est irréaliste. En pratique, comme les fournisseurs ont accès à tous les témoignages les concernant, rien ne les empêche de modifier les notes ou même de rajouter d'autres témoignages afin d'augmenter leur score de réputation, sans améliorer leur comportement.

4.2.1 Principe des boîtes aux lettres

Afin de résoudre ce problème, nous proposons d'utiliser une *boîte aux lettres* par fournisseur de service. À l'issue d'une transaction entre un client et un fournisseur de service, les deux participants déposent leur note concernant la transaction dans la boîte aux lettres du fournisseur, qui conserve le témoignage de la transaction. Ensuite, quand un client veut obtenir le score de réputation d'un fournisseur, il lui suffit d'interroger la boîte aux lettres du fournisseur pour obtenir les témoignages le concernant.

Cependant, une propriété importante des réseaux distribués à large échelle est son dynamisme (*churn*), c'est-à-dire que les agents se connectent et se déconnectent fréquemment du réseau avec des durées de connexion la plupart du temps faible [33]. On ne peut donc raisonnablement supposer qu'une boîte

aux lettres reste connectée en permanence pour conserver les témoignages d'un fournisseur. De plus, certaines boîtes aux lettres peuvent être corrompues et modifier les témoignages, soit pour faire croire qu'un fournisseur de service est bienveillant, soit qu'il est malveillant. Pour ces deux raisons, nous proposons d'utiliser plusieurs boîtes aux lettres par fournisseur. Cela permet de plus d'assurer aux clients que les témoignages sont intègres, sous l'hypothèse qu'il y a suffisamment de boîtes aux lettres bienveillantes.

4.2.2 Organisation des boîtes aux lettres

Comme nous venons de le voir, nous proposons d'affecter à chaque fournisseur un ensemble (grappe) de boîtes aux lettres afin de stocker les témoignages relatifs aux interactions de ce fournisseur de service avec les clients. Cet ensemble de boîtes aux lettres permet comme nous le verrons par la suite de garantir l'intégrité des témoignages, sous réserve qu'un quorum suffisant de boîtes aux lettres honnêtes existe.

L'architecture du réseau est modifiée en s'inspirant des travaux de Ravoaja et Anceaume [29] : des anneaux organisant les boîtes aux lettres par fournisseur sont utilisés à l'intérieur d'une DHT comme Chord, présentée au Chapitre 2. Les boîtes aux lettres d'un fournisseur de service sont les n plus proches voisins de ce fournisseur. Cette organisation est présentée à la Figure 4.7. Dans cette figure, AD est l'autorité de démarrage responsable de la génération des pseudonymes des clients, $FS_1, \ldots, FS_6$ sont des fournisseurs de service et $BaL(FS_2, j), j \in \{1, 2, 3\}$ est la grappe de boîtes aux lettres associées à FS_2.

L'utilisation de grappes de boîte aux lettres pose des problèmes de synchronisation comme dans tout système de stockage distribué : comment propager un nouveau témoignage pour s'assurer que toute boîte aux lettres bienveillante intègre le bon témoignage ?

De plus, s'il y a trop de boîtes aux lettres dans une grappe, il est trop coûteux pour un client de demander à toutes celles-ci quels sont les témoignages sur un fournisseur qu'elles stockent. Afin de résoudre ce problème de complexité, les clients n'interrogeront qu'un sous-ensemble réduit des boîtes aux lettres. Dans la suite, les notations suivantes seront utilisées :

I	Ensemble des indices des pseudonymes d'un client
J	Ensemble des indices des boîtes aux lettres d'un fournisseur
m	Nombre de boîtes aux lettres choisies par un client
J_m	Sous-ensemble de J de taille m

4.3 Protocole d'interaction entre un client et un fournisseur

Dans cette section, nous détaillons le protocole permettant à un agent A possédant les pseudonymes $p(A, i),\ i \in I$ d'interagir avec un fournisseur de service FS, associé aux boîtes aux lettres $BaL(FS, j)$ pour $j \in J$ (voir Figure 4.8). Nous supposons que les boîtes aux lettres sont honnêtes.

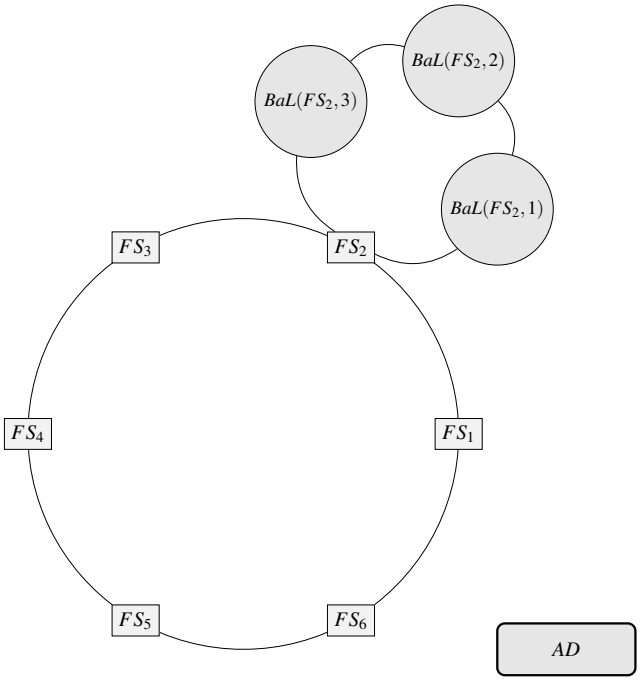


FIG. 4.7: Forme générale du réseau : structure en anneau des agents et grappes de boîtes aux lettres

Le protocole se découpe en quatre parties. La première permet à A de calculer le score de réputation de FS. Si après ce calcul, A n'est pas satisfait de la valeur du score de réputation de FS, le protocole s'arrête dès la fin de cette première partie. La seconde partie du protocole établit la communication permettant à la transaction entre un pseudonyme p de A et FS d'être engagée. La troisième partie décrit comment A et FS déposent leur témoignage sur les boîtes aux lettres. Enfin, la dernière partie décrit la synchronisation entre boîtes aux lettres.

4.3.1 Calcul de la réputation de FS par un agent A

L'agent A utilisant le pseudonyme $p(A, i)$ choisit aléatoirement un ensemble J_m de m boîtes aux lettres du fournisseur de service FS avec lesquelles il communiquera. Il réduit ainsi le nombre de messages envoyés en ne choisissant qu'une fraction des boîtes aux lettres disponibles. Si parmi les m boîtes aux lettres choisies, c ne répondent pas, $p(A, i)$ en choisit c aléatoirement parmi les boîtes aux lettres de J non encore choisies. Notons que dans la suite, l'existence d'un quorum de boîtes aux lettres disponibles jusqu'à la fin de la transaction est supposée.

Une fois ce choix effectué, A a besoin d'authentifier les boîtes aux lettres choisies, afin d'être sûr de leur identité, et de communiquer avec elles de manière confidentielle et intègre. Pour cela, il peut établir un tunnel TLS [10] avec elles. Une fois ces tunnels disponibles, le client demande à chacune des boîtes aux lettres les témoignages correspondant au fournisseur de service FS. Les

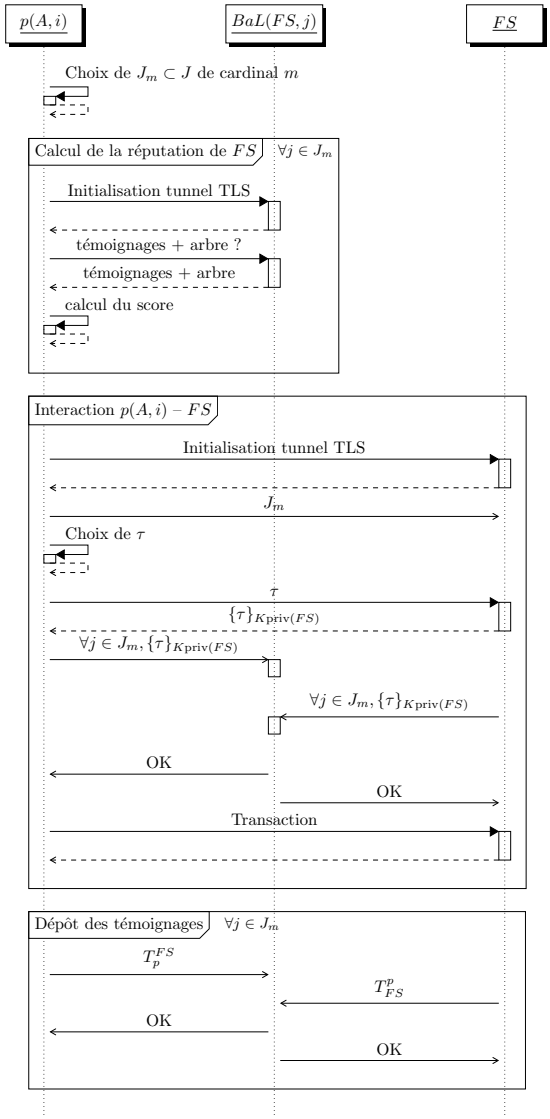


FIG. 4.8: Interaction entre un utilisateur A sous le pseudonyme $p(A,i)$, un fournisseur de service FS et ses boîtes aux lettres $BaL(FS,j)$ de calcul de score de réputation au dépôt des témoignages

boîtes aux lettres fournissent tous les témoignages, dans la limite d'un par pseudonyme (le plus frais s'il y en a plusieurs). Notons que A n'a besoin de connaître ni le pseudonyme ayant déposé le témoignage, ni l'identifiant de la transaction. Ces informations sont donc purgées des témoignages par les boîtes aux lettres. Une fois les témoignages expurgés récupérés, A combine les différentes notes et calcule le score de réputation de FS selon la méthode de calcul décrite précédemment. A choisit ensuite s'il désire ou non continuer l'interaction en fonction de la valeur de ce score.

4.3.2 Interaction entre FS et $p(A, i)$

Si A choisit d'interagir avec FS, il établit un tunnel TLS avec lui pour bénéficier comme précédemment d'un canal de communication sécurisé. Il informe ensuite FS des m boîtes aux lettres qu'il a choisi en lui envoyant J_m pour que les deux participants puissent communiquer avec ces mêmes boîtes aux lettres. $p(A, i)$ choisit un identifiant de transaction τ – par exemple en calculant une empreinte de $(p||FS||\text{timestamp})$[2] où timestamp représente le moment auquel cet échange a lieu – et le transmet à FS qui lui renvoie le même identifiant, signé avec sa clé privée. $p(A, i)$ informe ensuite les m boîtes aux lettres de l'interaction en leur envoyant cette signature. FS fait de même de son côté. Les boîtes aux lettres en accusent réception dès qu'elles ont reçu les deux exemplaires. Cela permet d'assurer aux deux participants que l'autre s'est lui aussi engagé à poursuivre l'interaction.

S'ensuit la transaction entre le client sous le pseudonyme $p(A, i)$, et le fournisseur de service FS. Nous ne faisons aucune hypothèse sur la nature ou le déroulement de cette transaction et la valeur et la date de la transaction sont supposées choisies par les deux participants.

4.3.3 Dépôt des témoignages de FS et $p(A, i)$ sur les boîtes aux lettres de J_m

Une fois la transaction terminée, les participants déposent leur note sur les boîtes aux lettres de J_m en utilisant les tunnels de communication crées en première partie du protocole. Les boîtes aux lettres les informent quand les deux témoignages sont récupérés. En d'autres termes, si après un certain temps d'attente t_a, les boîtes aux lettres n'ont reçu qu'un seul des deux témoignages, elles considèrent le témoignage manquant comme étant neutre vis à vis de la transaction. Si le témoignage manquant est celui du client alors elles le considèrent comme un avis positif à propos de FS. Si le témoignage manquant est celui de FS alors elles le considèrent comme un avis négatif à propos de la transaction. Après ce temps t_a, les témoignages sont rendus disponibles à tous.

4.3.4 Synchronisation des boîtes aux lettres

Afin d'assurer une synchronisation des boîtes aux lettres, nous proposons d'appliquer le problème de l'accord byzantin [21] à l'ajout d'un témoignage pour éviter d'intégrer un témoignage fallacieux. En effet, les boîtes aux lettres peuvent être vues comme des processus pouvant avoir un comportement byzantin : les boîtes aux lettres peuvent par exemple subir des déconnexions du réseau ou colporter de faux témoignages. Cette application du consensus byzantin dans les DHT a déjà été proposée par Fedotova *et al.* [13] pour l'algorithme du système EigenTrust [20]. Nous proposons de l'utiliser pour synchroniser les boîtes aux lettres et s'assurer qu'une d'entre elles ne bloque pas l'arrivée d'un nouveau message. Pour l'instant, nous supposons qu'une boîte aux lettres ne peut pas se déconnecter du réseau au milieu d'une interaction : une interaction est atomique. Nous supposons également que le système

[2] $||$ est l'opérateur de concaténation.

est partiellement synchrone, c'est-à-dire qu'il existe une borne connue sur le temps de transmission d'un message.

Un problème demeure concernant l'initialisation de l'algorithme : en effet, initialement, seulement m boîtes aux lettres – celles de J_m, choisies par le client – connaissent le témoignage à ajouter. Il faut ainsi commencer par une étape de diffusion : chaque boîte aux lettres $BaL(FS, j_m), j_m \in J_m$ annonce le témoignage reçu à toutes les autres. Ainsi, chaque boîte aux lettres a reçu j_m messages en un temps fini grâce aux hypothèses de synchronisme et d'atomicité. Localement, chaque boîte aux lettres choisit le témoignage le plus fréquent si son nombre d'occurrences est supérieur à $\lfloor m/2 \rfloor + 1$. Si jamais il n'y a pas de témoignage plus fréquent ou que son nombre d'occurrences n'est pas assez grand, la boîte aux lettres ne choisit pas de témoignage. Finalement, les boîtes aux lettres exécutent un algorithme de consensus en proposant le témoignage choisi. Si jamais une boîte aux lettres n'a pas choisi de témoignage, elle propose une valeur « témoignage vide ». L'hypothèse de synchronisme nous permet d'utiliser un des algorithmes présenté par Dwork *et al.* [12].

De cette façon, la synchronisation des boîtes aux lettres est assurée à tout instant tant qu'elles sont synchronisées à un instant initial et qu'elles ne quittent pas le réseau. Il faut donc assurer que dès qu'une boîte aux lettres s'insère dans J, elle soit synchronisée, et gérer les concurrences qui peuvent survenir. À cet effet, nous proposons d'utiliser un compteur de témoignages ainsi qu'une zone tampon pour les nouvelles boîtes aux lettres. Le compteur de témoignages, initialement proposé dans [4], permet à une boîte aux lettres de savoir rapidement si elle est à jour ou s'il lui manque des témoignages. Nous proposons d'utiliser la zone tampon de la manière suivante : à son arrivée, une nouvelle boîte aux lettres se place dedans. Elle ne peut être choisie comme boîte aux lettres par un client, mais elle communique normalement avec les autres boîtes et ajoute les nouveaux témoignages. En même temps, elle demande la liste des témoignages à m boîtes aux lettres. Une fois ceux-ci récupérés, elle sort de la zone tampon et est disponible pour les clients.

Chapitre 5

Attaques sur le système et la vie privée

Le protocole présenté à la Section 4.3 permet à un client de récupérer les témoignages d'un fournisseur de service, de calculer son score de réputation, d'effectuer une transaction avec lui et finalement d'émettre un témoignage. Dans ce chapitre, nous montrons que ce protocole permet à notre système d'être robuste (cf. Propriété 1) et que la vie privée des clients est préservée (cf. Propriété 2).

5.1 Robustesse du système de réputation

La première propriété concernant la robustesse du système de réputation est découpée en lemmes, spécifiques au système de réputation proposé. La robustesse du système de réputation est garantie si les lemmes suivant le sont :

Lemme 1 (Témoignages)**.** *Si le nombre de boîtes aux lettres malveillantes en collusion ne dépasse pas une certaine proportion, un témoignage ne peut être inventé ou supprimé de la liste, et l'intégrité des témoignages est garantie avec une grande probabilité.*

Lemme 2 (Non-répudiation d'une transaction)**.** *Si le nombre de boîtes aux lettres malveillantes en collusion ne dépasse pas une certaine proportion, un agent ne peut essayer de répudier une transaction sans que cela ne lui porte préjudice – c'est-à-dire que si cet agent est client, l'impact sera positif pour le fournisseur et négatif pour le fournisseur si cet agent est fournisseur.*

En effet, si ces deux lemmes sont vrais, alors les témoignages obtenus par un client sont *les* témoignages émis par des clients passés. C'est-à-dire qu'aucun témoignage n'a été rajouté à cette liste, aucun n'a été enlevé, et ils n'ont pas été modifiés. Ceci est donné par le Lemme 1. De plus, toutes les transactions sont prises en compte – dans la limite d'une par pseudonyme, la plus fraîche. Ceci est garanti par le Lemme 2.

Il ne reste donc qu'à montrer que le score de réputation reflète le comportement du fournisseur de service malgré un certain nombre de témoignages malveillants – c'est-à-dire médisants dans le cas d'un fournisseur de service malhonnête. La Proposition 1 est introduite à cet effet. Pour cette proposition,

le fournisseur de service FS est supposé se comporter honnêtement avec une probabilité P_h et correctement avec une probabilité P_c. Le client cherche à obtenir le score de réputation de FS avec une précision ε_c en conformité et ε_h en honnêteté. Lors de la collecte des témoignages, il dénombre $\overline{\rho}_{h,+}$ témoignages positifs et $\overline{\rho}_{h,-}$ témoignages négatifs concernant l'honnêteté de FS, et $\overline{\rho}_{c,+}$ et $\overline{\rho}_{c,-}$ pour sa conformité.

Proposition 1 (Précision du score de réputation)**.** *Pour un nombre de témoignages malveillants et une précision voulue donnés, il est possible de calculer un score de réputation représentatif du comportement réel d'un fournisseur de service FS en augmentant le nombre de témoignages pris en compte. Cela se traduit par l'équation*

$$\forall \varepsilon_c,\ \varepsilon_h > 0,\ \exists N_0 \mid \text{card}(T_{FS}) > N_0 \Rightarrow \begin{cases} |P_h - \dfrac{\overline{\rho}_{h,+}}{\overline{\rho}_{h,+} + \overline{\rho}_{h,-}}| < \varepsilon_h \\ |P_c - \dfrac{\overline{\rho}_{c,+}}{\overline{\rho}_{c,+} + \overline{\rho}_{c,-}}| < \varepsilon_c \end{cases} \tag{5.1}$$

Démonstration. Afin de vérifier que cette proposition est vraie, une hypothèse non-limitative est supposée : un témoignage vaut 1 si le fournisseur de service est honnête lors d'une transaction, 0 sinon. Cela simplifie les calculs et rend les témoignages parfaitement symétriques pour l'honnêteté et pour la conformité. On considère un facteur de longévité $\lambda \in]0;1[$. On considère également qu'un témoignage est posté par unité de temps, ainsi le premier témoignage est vieilli de λ et le n-ième de λ^{n_t}.

Soient $c_t > 0$ le nombre de témoignages malveillants et $\varepsilon > 0$ la précision dans le calcul du score de réputation souhaitée. L'objectif est de trouver N_0 suffisamment grand pour que, dès qu'il y ait plus de N_0 témoignages concernant FS – dont c_t malveillants –, le score soit précis à ε près. L'espérance du score de réputation du fournisseur de service sans témoignages malveillants est notée E, et $\widehat{E}$ avec les témoignages malveillants.

Pour n_t témoignages, le pire cas est celui où les c_t témoignages malveillants se réfèrent à des interactions où le fournisseur de service s'est bien comporté. Cela augmente le nombre de témoignages négatifs. Ce comportement est d'autant plus accentué que les témoignages positifs sont anciens. Le pire cas est donc celui où le fournisseur de service a commencé par se comporter honnêtement $P_h n_t - c_t$ fois, puis malhonnêtement $(1 - P_h)n_t$ fois et finalement où les c_t derniers témoignages sont fallacieux.

La somme des témoignages positifs vieillis est la suivante :

$$\sum_{k=n_t(1-P_h)+c_t+1}^{n_t} \lambda^k$$

La différence entre les deux scores de réputations est la suivante :

$$
\begin{aligned}
|E - \widehat{E}| &= \frac{1}{\sum_{k=1}^{n_t} \lambda^k} \left| \sum_{i=n_t(1-P_h)+c_t+1}^{n_t} \lambda^i - \sum_{j=n_t(1-P_h)+1}^{n_t} \lambda^j \right| && (5.2)\\
&= \frac{1}{\lambda \frac{1-\lambda^{n_t}}{1-\lambda}} \left| \lambda^{n_t(1-P_h)+c_t+1} \frac{1-\lambda^{P_h n_t - c_t}}{1-\lambda} - \lambda^{n_t(1-P_h)+1} \frac{1-\lambda^{P_h n_t}}{1-\lambda} \right| \\
&= \frac{1}{1-\lambda^{n_t}} \left| \lambda^{n_t(1-P_h)+c_t} (1-\lambda^{P_h n_t - c_t}) - \lambda^{n_t(1-P_h)} (1-\lambda^{P_h n_t}) \right| \\
&= \frac{\lambda^{n_t(1-P_h)}}{1-\lambda^{n_t}} |\lambda^{c_t} - 1| && (5.3)\\
&\underset{n_t \to +\infty}{\sim} \lambda^{n_t(1-P_h)} (1-\lambda^{c_t}) \\
&\underset{n_t \to +\infty}{\to} 0
\end{aligned}
$$

Ainsi on peut augmenter la précision en augmentant le nombre de témoignages considérés. Les hypothèses simplificatrices considérées semblent ne pas impacter cette augmentation de la précision outre mesure. □

Ainsi, en augmentant infiniment le nombre de témoignages, l'erreur sur le score calculé se rapproche de zéro. Cependant, cette preuve ne donne pas d'évolution quantitative de la précision. Afin de s'en rendre compte, nous modélisons les clients comme ayant un « biais », c'est-à-dire modifiant les notes des témoignages en leur ajoutant un nombre $b \in [-0.1; 0.1]$ – celle-ci est ramenée à 0 ou 1 si elle dépassait d'un côté ou de l'autre. On regarde ensuite la différence de score calculé en ajoutant 5 clients médisants, c'est-à-dire déposant un témoignage nul. Pour un nombre de clients variant dans $\{20; 30; 50; 100\}$, nous calculons le score de réputation du fournisseur de service en prenant en compte tous les témoignages, une fois en comprenant le biais et les clients médisants et une autre fois, sans biais ni médisances. La figure 5.1 montre la distribution des résultats dans une « boîte à moustache » – c'est à dire les 5^e, 25^e, 50^e, 75^e et 95^e centiles de la distribution – pour chaque nombre de clients en prenant 100 simulations différentes en compte.

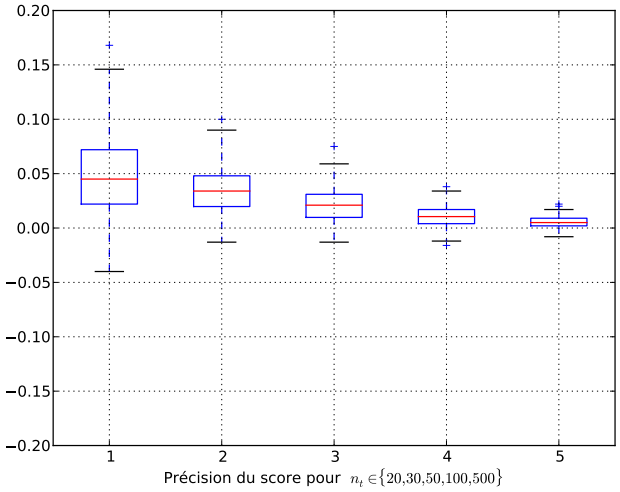


FIG. 5.1: Boîte à moustache des écarts entre le score calculé et le comportement d'un fournisseur de service, pour 100 simulations

5.1.1 Preuve du Lemme 1

Le Lemme 1 consiste à montrer qu'il est très peu probable qu'une collusion de boîtes aux lettres parvienne à faire croire à un client que les témoignages concernant un fournisseur de service soient autres que les véritables.

Preuve du Lemme 1. Tout d'abord, il convient de montrer que le client communique avec m véritables boîtes aux lettres du fournisseur de service. En effet, il établit avec elles un tunnel TLS, ce qui garantit l'authentification de ces boîtes aux lettres et lui permet d'effectuer des échanges confidentiels et intègres, ce qui prévient toute tentative d'usurpation d'identité ou de rejeu de messages par un attaquant potentiel.

Ensuite, le client choisit m boîtes aux lettres afin d'obtenir les témoignages concernant le fournisseur. Il faut veiller à ce que la probabilité d'obtenir une majorité de boîtes aux lettres corrompues, c'est-à-dire donnant des témoignages fallacieux, soit très faible. Pour cela, on considère que les boîtes aux lettres corrompues sont réparties aléatoirement. Le choix de m boîtes aux lettres parmi n revient à un tirage sans remises de m boîtes aux lettres parmi n avec deux types de boîtes : c corrompues et $n - c$ honnêtes. La probabilité d'obtenir k boîtes aux lettres corrompues en en tirant m est

$$P(X = k) = \frac{\binom{c}{k}\binom{n-c}{m-k}}{\binom{n}{m}}$$

En effet, il y a $\binom{c}{k}$ tirages possibles de k boîtes corrompues, et $\binom{n-c}{m-k}$ tirages possibles de $m - k$ boîtes honnêtes, parmi $\binom{n}{m}$ tirages possibles. La probabilité

d'obtenir un quorum de boîtes aux lettre corrompues est la somme des probabilités d'obtention de k boîtes, pour k au moins égal à la majorité, c'est-à-dire :

$$P_{\text{coal}} = \sum_{k=\lfloor m/2 \rfloor + 1}^{m} P(X = k)$$

Les résultats sont présentés à la figure 5.2, pour $n = 20$ et $n = 200$ et une proportion de boîtes aux lettres corrompues $c = 5, 10, 15$ et 25%.

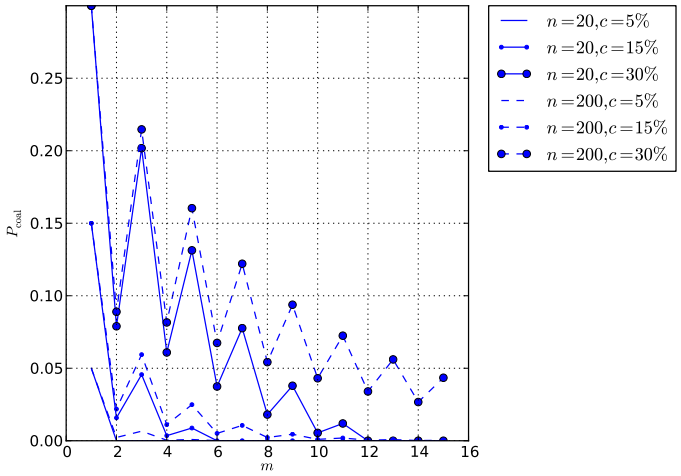


FIG. 5.2: Probabilité d'obtenir une coalition de boîtes aux lettres $P_{\text{coal}} = f(m)$ avec un schéma de sélection aléatoire

On remarque que pour un faible nombre de boîtes choisies, la probabilité d'obtenir une coalition peut être très élevée, et est indépendante du nombre de boîtes aux lettres total. Plus on augmente le nombre de boîtes nécessaires à l'élaboration du consensus, plus elle diminue et, pour $n = 200$ et 25% de boîtes aux lettres corrompues (soit 50), prendre 9 boîtes suffit à descendre en dessous des 5% de risque, ce qui permet de détecter toute collusion de taille réduite. Si jamais une telle collusion est détectée, le client peut décider d'ignorer les boîtes aux lettres de la collusion et faire confiance aux témoignages stockés par les autres boîtes.

□

5.1.2 Preuve du Lemme 2

Afin de ne pas augmenter le score d'un fournisseur de service, un client peut réfuter une transaction, c'est-à-dire faire comme s'il n'y avait eu aucun échange une fois que la transaction est finie. De la même façon, un fournisseur de service peut réfuter une transaction où il s'est comporté malhonnêtement afin de ne pas voir son score de réputation diminuer. Pour faire disparaître ces deux comportements, il suffit de pénaliser l'agent à sa source :

Preuve du Lemme 2. Si $p(A, i)$ interagit avec un fournisseur FS bienveillant mais ne veut pas augmenter la réputation de FS, il peut se déconnecter volontairement du réseau une fois la transaction effectuée mais avant d'avoir déposé son témoignage. Cependant, si la transaction est effecutée, la boîte aux lettres a reçu l'engagement du client sur la transaction – $\{\tau\}_{K\text{priv}(FS)}$ – et, après un certain temps t_a passé à l'attendre, la boîte aux lettres attribue d'elle-même un témoignage positif pour FS. Un client est ainsi incité à témoigner lorsqu'une transaction s'est mal déroulée.

De même, si FS se déconnecte après s'être mal comporté sans émettre de témoignage, la boîte aux lettres attribue un témoignage négatif sur la transaction, incitant ainsi FS à témoigner. L'avantage de cette méthode est qu'elle pallie toutes les tentatives de réfutation provenant d'un fournisseur de service, prouvant ainsi le lemme 2.

Par contre, les déconnexions (involontaires) du réseau sont également détectées comme des tentatives de réfutation. □

5.2 Respect de la vie privée

D'après la propriété 2, nous considérons que la vie privée d'un client est préservée si trois conditions sont respectées : le pseudonymat des clients, la non-traçabilité d'une identité et d'un pseudonyme et enfin la non-traçabilité de deux pseudonymes.

Dans notre proposition, un client n'utilise jamais son identité réelle. En effet, avant toute interaction il génère des pseudonymes qu'il fait certifier par l'autorité de démarrage. Ces pseudonymes sont ensuite utilisés pour toute interaction. Ainsi, le *pseudonymat* des agents est assuré, de même que la *non-traçabilité* d'une identité et d'un pseudonyme.

La propriété la plus intéressante est la non-traçabilité des pseudonymes. En effet, celle-ci garantit que l'on ne peut pas savoir si deux pseudonymes différents appartiennent à la même identité ou pas. Il existe de nombreuses attaques par inférence, où l'on s'intéresse à des comportements révélateurs. Un exemple significatif est celui présenté par Barbaro et Zeller [7] : en 2006, après le dévoilement par AOL de données de recherches des utilisateurs sur son moteur de recherche, certaines personnes furent retrouvées. Notamment, Thelma Arnold, grâce à quelques unes de ses recherches révélatrices : recherches généalogiques, de commerces de proximité, etc.

Pour l'instant, l'impact de telles attaques sur notre système n'a pas été étudié, mais nous supposons qu'elles peuvent être menées en comparant les différents éléments des témoignages : soit par des habitudes temporelles, grâce aux dates d'émission des témoignages, soit par le comportement d'un client, c'est-à-dire en s'intéressant à la portée des notes émises. En effet, un client « optimiste » a tendance à surnoter toute transaction, tandis qu'un râleur a tendance à sous-noter toute transaction.

Chapitre 6

Conclusion et travaux futurs

Dans ce rapport, nous avons commencé par présenter une taxonomie originale des systèmes de réputation suivant quatre éléments : stockage des témoignages, sélection des témoins, collecte des témoignages et calcul du score de réputation. Nous avons ensuite expliqué quelles sont les principales attaques contre un système de réputation et son réseau sous-jacent. Le fonctionnement du seul système de réputation préservant la vie privée à notre connaissance a ensuite été détaillé. Notre proposition, un système de réputation distribué respectant la vie privée des utilisateurs et robuste aux attaques, a ensuite été présentée, en supposant dans un premier temps que les fournisseurs de service ne cherchent pas à modifier les témoignages. Cette hypothèse a ensuite été supprimée en introduisant des grappes de nœuds appelés *boîtes aux lettres* dont le rôle est de stocker les témoignages. Nous proposons un protocole d'interaction, combinant à la fois des outils cryptographiques standards et l'utilisation des boîtes aux lettres afin de permettre la robustesse du système. Finalement, nous prouvons la robustesse du système ainsi que la plupart des propriétés de respect de la vie privée souhaitées.

Ayant à notre disposition encore quelques mois de stage pour poursuivre ce travail, nous pensons tout d'abord nous intéresser à la non-traçabilité des pseudonymes pour montrer qu'un attaquant ne peut inférer un lien entre plusieurs pseudonymes à partir des connaissances dont il dispose. L'étape de modulation proposée dans le calcul du score ne permettant d'inciter que le fournisseur de service à émettre une note réaliste de la transaction, nous verrons s'il est possible d'utiliser les travaux de Michiardi et Molva [23], proposant des mécanismes d'incitation à la coopération dans des réseaux mobiles ad-hoc en utilisant la théorie des jeux. Finalement, nous nous attacherons à développer une simulation permettant de mesurer l'impact du nombre de boîtes aux lettres malveillantes sur le système.

Table des figures

Bibliographie

[1] eBay. http://www.ebay.com.

[2] Page Wikipedia Google bomb. http://en.wikipedia.org/wiki/Google_bomb.

[3] A. Abdul-Rahman and S. Hailes. Supporting trust in virtual communities. In *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, pages 9–pp. IEEE, 2000.

[4] R. Akbarinia, M. Tlili, E. Pacitti, P. Valduriez, and A. Lima. Continuous Timestamping for efficient Replication Management in DHTs. *Data Management in Grid and Peer-to-Peer Systems*, pages 38–49, 2011.

[5] E. Androulaki, S. Choi, S. Bellovin, and T. Malkin. Reputation systems for anonymous networks. In *Privacy Enhancing Technologies*, pages 202–218. Springer, 2008.

[6] M.S. Artigas, P.G. Lopez, J.P. Ahullo, and A.F.G. Skarmeta. Cyclone : A novel design schema for hierarchical dhts. In *Fifth IEEE International Conference on Peer-to-Peer Computing (P2P 2005)*, pages 49–56. IEEE, 2005.

[7] M. Barbaro and T. Zeller. A face is exposed for AOL searcher no. 4417749. *The New York Times*, 2006.

[8] N. Borisov. Computational puzzles as sybil defenses. In *Sixth IEEE International Conference onPeer-to-Peer Computing (P2P 2006)*, pages 171–176. IEEE, 2006.

[9] E. Carrara and G. Hogben. Reputation-based systems : a security analysis. *ENISA Position Paper*, 2007.

[10] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), August 2008. Updated by RFCs 5746, 5878, 6176.

[11] J. Douceur. The sybil attack. *Peer-to-peer Systems*, pages 251–260, 2002.

[12] C. Dwork, N. Lynch, and L. Stockmeyer. Consensus in the presence of partial synchrony. *Journal of the ACM (JACM)*, 35(2) :288–323, 1988.

[13] N. Fedotova, G. Orzetti, L. Veltri, and A. Zaccagnini. Byzantine agreement for reputation management in DHT-based peer-to-peer networks. In *International Conference on Telecommunications (ICT 2008)*, pages 1–6. IEEE, 2008.

[14] A. Fiat, J. Saia, and M. Young. Making Chord robust to byzantine attacks. *Algorithms–ESA 2005*, pages 803–814, 2005.

[15] J. Fiser. Perceptual learning and representational learning in humans and animals. *Learning & Behavior*, 37(2) :141–153, 2009.

[16] K. Hoffman, D. Zage, and C. Nita-Rotaru. A survey of attack and defense techniques for reputation systems. *ACM Computing Surveys (CSUR)*, 42(1) :1, 2009.

[17] A. Josang. Trust-based decision making for electronic transactions. In *Proceedings of the Fourth Nordic Workshop on Secure Computer Systems (NORDSEC'99)*, pages 496–502, 1999.

[18] A. Jøsang and R. Ismail. The beta reputation system. In *Proceedings of the 15th Bled Electronic Commerce Conference*, 2002.

[19] A. Josang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2) :618–644, 2007.

[20] S.D. Kamvar, M.T. Schlosser, and H. Garcia-Molina. The EigenTrust algorithm for reputation management in P2P networks. In *Proceedings of the 12th international conference on World Wide Web*, pages 640–651. ACM, 2003.

[21] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3) :382–401, 1982.

[22] S. Marti and H. Garcia-Molina. Taxonomy of trust : Categorizing P2P reputation systems. *Computer Networks*, 50(4) :472–484, 2006.

[23] Pietro Michiardi and Refik Molva. A game theoretical approach to evaluate cooperation enforcement mechanisms in mobile ad hoc networks. In *In Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, pages 3–5, 2003.

[24] S. Nakamoto. Bitcoin : A peer-to-peer electronic cash system, 2009.

[25] E. Pavlov, J. Rosenschein, and Z. Topol. Supporting privacy in decentralized additive reputation systems. *Trust Management*, pages 108–119, 2004.

[26] A. Paz. Introduction to probabilistic automata. 1971.

[27] Andreas Pfitzmann and Marit Hansen. A terminology for talking about privacy by data minimization : Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. http ://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf, Août 2010. v0.34.

[28] A. Ravoaja. *Mécanismes et architectures P2P robustes et incitatifs pour la réputation*. PhD thesis, Université Rennes-1, 2008.

[29] A. Ravoaja and E. Anceaume. Storm : A secure overlay for p2p reputation management. In *Proceedings of the International Conference on Self-Autonomous and Self-Organizing Systems (SASO)*, 2007.

[30] T. Reidemeister, K. Böhm, E. Buchmann, and P.A.S. Ward. Man-in-the-middle attacks in distributed hash-tables. *IEEE Journal on Selected Areas in Communication*, 2006.

[31] A. Saroliya and V. Shrivastava. Security problems and their upshots in routing protocols of dht based overlay networks. 2005.

[32] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord : A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the ACM SIGCOMM '01 Conference*, San Diego, California, August 2001.

[33] D. Stutzbach and R. Rejaie. Understanding churn in peer-to-peer networks. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 189–202. ACM, 2006.

[34] P.F. Syverson, D.M. Goldschlag, and M.G. Reed. Anonymous connections and onion routing. In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, pages 44–54. IEEE, 1997.

[35] G. Urdaneta, G. Pierre, and M.V. Steen. A survey of DHT security techniques. *ACM Computing Surveys (CSUR)*, 43(2) :8, 2011.

[36] A. Whitby, A. Jøsang, and J. Indulska. Filtering out unfair ratings in bayesian reputation systems. In *Proceedings of the 7th International Workshop on Trust in Agent Societies*, 2004.