Secured trust and reputation system : analysis of malicious behaviors and optimization

Amira Bradai

Ecole Doctorale EDITE
Thèse présentée pour l'obtention du diplôme de Docteur
conjoint télécom SudParis et université Pierre et Marie Curie

Spécialité:

Informatique et Télécommunications

Gestion de la confiance et la réputation sécurisée : Analyse des attaques possibles et optimisation

Par

Amira Bradai

Soutenue le 29 Septembre 2014 devant le jury composé de :

Mme Veronique VEQUE	Rapporteur	Université Paris Sud
Mr Ken CHEN	Rapporteur	Institut GALILEE, Univ. Paris 13
Mr Pascal LORENZ	Examinateur	Université Haute Alsace IUT
Mr Vincent ROCA	Examinateur	INRIA, Grenoble
Mr Guy PUJOLLE	Examinateur	UPMC, Paris 6
Mr Hossam AFIFI	Directeur de thèse	Télécom SudParis

Thése 2014TELE0019

Universite Pierre et Marie Curie
Institut Mines-Telecom
Telecom SudParis

Thesis Report

To obtain the degree of

Docteur d'université

Subject

Secured trust and reputation system: analysis of malicious behaviors and optimization

Presented by

Amira Bradai

Mme Veronique VEQUE	Reviewer
Mr Ken CHEN	Reviewer
Mr Pascal LORENZ	Examiner
Mr Vincent ROCA	Examiner
Mr Guy PUJOLLE	Examiner
Mr Hossam AFIFI	Thesis Director

To my parents Foued and Monia,
I am especially thankful for your love, your understanding and your continuous support. Thanks for always believing in me.
To my husband Chiheb eddine
I am particularly indebted for your unconditional trust, support and sincere love.
To my sister Youmna, Imen
To my brother Walid
To my family in law :Bachir, Sabria, Yesmine and Safwen
Thanks for being always in my side during difficult moments!
I dedicate this work also to **my baby in my belly**

Acknowledgement

Throughout my Phd study period in the SAMOVAR laboratory at Telecom SudParis, many people have kindly provided me with their help and unlimited support. It is a pleasure to convey my most profound gratitude to them all and I am particularly indebted to the following people.

First and foremost, I want to express my deep and sincere gratitude to my supervisor, Prof. Hossam Afifi for his continuous support and his constant guidance during my PhD study years in the SAMOVAR laboratory. Your encouragement and personal guidance have provided a good basis for the present thesis. Your perpetual energy and enthusiasm in research had motivated me so much. I am very fortunate to have you as a supervisor. Thank you for everything, it was a truly great experience working with you !

Second, I am indebted to Prof. Walid Ben-Ameur for his warmly reception and insightful discussions and instructions during my research study. Under his supervision the quality of my work has constantly improved, not to mention the invaluable support and countless contributions to this work. Thank you for having oriented and supported me with patience and encouragements, for your great sense of responsibility and professionalism.

I wish to thank also the members of the dissertation jury for accepting the invitation and their valuable time and feedback. My special appreciation goes to professor Guy Pujole, professor Veronique Veque, professor Pascal Lorenz, professor Ken Chen and Dr.Vincent Roca.

I would like also to thank the staff of Telecom SudParis; particularly Dr.Abdallah M'hamed. Many thanks go to Mme Valérie Mateus and Françoise Abad who take care of my demands on the administrative procedures.

Many thanks to all my colleagues and friends inside Telecom SudParis and outside also for the excellent and truly enjoyable ambiance. My warmest thanks go to: Emad Abd-Elrahman, Ahmed Soua, Dorsaf Zekri, Rahma Yengui, Salma Rebai, Ines Fakhfakh, kaouther Drira, Houda Jmila, Nadia Masmoudi, Chayma Ghribi, Khalifa Toumi, Fadwa Dalel, Adel Mounir, Rachit Agarwal and Mohamed Abid. I am grateful to all of you, for your encouragements, support and for the fun moments I have shared with you !

My endless thanks to my husband, my sister, for the support

Last, I want to express my gratitude to my family for their unconditional love,

support, and trust.

Abstract

Reputation mechanisms offer a novel and effective way of ensuring the necessary level of trust which is essential to the functioning of any critical system. They collect information about the history (i.e., past transactions) of participants and make public their reputation. Prospective participants guide their decisions by considering reputation information, and thus make more informative choices. Online reputation mechanisms enjoy huge success. They are present in most e-commerce sites available today, and are seriously taken into consideration by human users. Existing reputation systems were conceived with the assumption that users will share feedback honestly.

But, such systems like those in peer to peer are generally compromise of malicious users. This leads to the problem in cooperation, aggregation and evaluation. Some users want to use resources from network but do not want to contribute back to the network. Others manipulate the evaluations of trust and provide wrong estimation. We have recently seen increasing evidence that some users strategically manipulate their reports and behave maliciously. For proper protecting against those users, some kind of reputation management system is required. In some system, a trusted third entity exist and can aggregate the information. However, peer-to-peer networks don't have any central control or repository. Large size of distributed and hybrid networks makes the reputation management more challenging task. Hence reputation management system should perform all the tasks in distributed fashion. When these kind of systems are implemented, peers try to deceive them to take maximum advantage.

This thesis describes ways of making reputation mechanisms more trustworthy and optimized by providing defense mechanism and analysis. Different kinds of malicious behaviors exist and for each one, we present a complete analysis, simulation and a real use case example in distributed and non distributed way.

After the analysis of the related state of art, we first propose a reputation based solution for the hybrid cloud computing. As an attack model is proposed and described, we built a defense mechanism based on credibility parameter. We show that our methods are more efficient compared to the well known EigenTrust and the average approach in cloud computing. As a validation of this first contribution, we prove three results in two use cases showing the advantage of our models:

- in hybrid cloud computing, the allocation of jobs is better and more efficient with trust model.

- in hybrid cloud computing, jobs' loss (jobs affected to malicious peers) is less important when we consider the trust model.
- when the selection of sources are based on our trust model, the number of inauthentic files downloaded in peer to peer network is reduced

After validation, we proposed as the second contribution the use of game theory as a way of optimization in distributed intrusion detection. The problem was that some peers can optimize some resources while evaluating other peers and participating in detection.

In the third and the fourth part, we are interested in the distributed architecture. To manage the reputation system, we proposed a gossip based aggregation model. Then, we validate our analytical model. After that, we analysis our proposal in the presence of stubborn peers that want to spread misinformation. A second manipulation of reputation can occur. So in the last part, we simulate the impact of forceful peers on our reputation model aggregation. Two forceful peers can adopt some strategies based on some information about the network. We are showing that in different graphes (random, scale free..), the impact is different. It is related to the topology, the strategy adopted (random, based on the degree ...) and the distribution of the peers in the network.

Key words: trust management, reputation system, aggregation, gossip, malicious, stubborn, forceful, byzantine, game theory

Résumé

Les mécanismes de réputation offrent un nouveau moyen efficace pour assurer le niveau de confiance qui est indispensable au bon fonctionnement de tout système critique. Ce fonctionnement consiste à collecter les informations sur l'historique des participants et rendre public leur réputation. Grâce à ces informations, le système oriente les décisions afin d'optimiser le choix des solutions les plus sécurisées. Des mécanismes de réputation en ligne sont présents dans la plupart des sites e-commerce disponibles aujourd'hui. Les systèmes existants ont été conçus avec l'hypothèse que les utilisateurs partagent les informations honnêtement.

Mais, beaucoup de systèmes de réputation sont en général un sujet d'attaque par les utilisateurs malveillants. L'attaque peut affecter la coopération, l'agrégation et l'évaluation. Certains utilisateurs veulent utiliser les ressources du réseau, mais ne veulent pas contribuer en retour. D'autres manipulent les évaluations de la confiance et donnent une mauvaise estimation. De nos jours, on constate de plus en plus que certains utilisateurs manipulent stratégiquement leurs évaluations et se comportent d'une façon malhonnête. Pour une protection adéquate contre ces utilisateurs, un système sécurisé pour la gestion de la réputation est nécessaire. Dans notre système, une entité centrale existe et peut agréger les informations . Cependant, les réseaux pair à pair ne disposent pas de contrôle centralisé et de référentiel commun ce qui rend la tâche plus difficile. Par conséquent, le système de gestion de la réputation doit effectuer toutes les tâches de manière distribuée. Lorsque ce genre de systèmes est mis en oeuvre, les pairs essaient de plus en plus de manipuler les informations.

Cette thèse décrit les moyens permettant de rendre les mécanismes de réputation plus sécurisés en analysant les risques et en fournissant un mécanisme de défense. Différents types de comportements malveillants existent et pour chacun d'eux, nous présentons une analyse complète, des simulations et un exemple d'utilisation réel.

Après l'analyse de l'état de l'art, nous avons proposé tout d'abord un système de réputation et de la confiance sécurisé. Ensuite, nous avons élaboré le mécanisme de défense en se basant sur le paramètre de crédibilité après avoir étudié et proposé un modèle d'attaque. Nous avons montré que nos méthodes sont plus efficaces par rapport à EigenTrust, un modèle bien connu, et un autre utilisé dans les nuages informatiques. Pour la validation de cette première contribution, nous avons montré à l'aide des simulations trois résultats dans deux cas d'utilisation :

- La répartition des tâches dans les nuages informatiques est meilleure et plus

efficace avec notre modèle de confiance.

- La perte des tâches (soumission à des noeuds malicieux) dans les nuages informatiques est moins importante lorsque nous considérons le modèle de confiance.
- Le téléchargement des fichiers non authentiques dans le réseau pair à pair est moins important si on considère nos algorithmes dans la simulation de partage de fichiers.

Après cette validation, nous avons proposé comme deuxième contribution l'utilisation de la théorie des jeux afin d'assurer l'optimisation dans la détection des noeuds malicieux en mode distribué. Le problème est que certains pairs peuvent optimiser des ressources tout en évaluant les autres.

Dans les dernières contributions, nous avons proposé une agrégation de la réputation basée sur "gossip". Puis, nous avons analysé notre proposition validée en présence de pairs têtus qui veulent propager la désinformation et le désaccord. Aussi, une seconde manipulation de la réputation peut se produire : il s'agit des pairs stratégiques qui influencent les opinions des autres. Pour les mettre en évidence, nous avons confronté deux pairs qui peuvent adopter des stratégies différentes. Ces stratégies sont proposées en prenant en considération les informations sur le réseaux comme le voisinage. Nous montrons que dans différents graphes, les performances et l'impact sont différents et dépendent de la stratégie, du type de graphe et du budget des pairs qui s'affrontent.

Mots clès: la gestion de la confiance, la réputation, agrégation, malveillant, stratégies, têtus, théorie des jeux, byzantin

Thesis Publications

Journal Paper

- **Amira Bradai**, Walid Ben-Ameur, and Hossam Afifi, "Secure Reputation and Trust System" . Submitted to Dependable and Secure Computing, IEEE

Conference Papers

1. Amira Bradai, and Hossam Afifi,"Enforcing Trust-Based Intrusion Detection in Cloud Computing Using Algebraic Methods", CyberC 2012, Sanya ,Chine
2. Amira Bradai, Emad abdelrahmen, and Hossam Afifi, " Privacy Aware Nomadic Service For Personalized IPTV ", ICSNC 2013, Venise, Italy
3. Amira Bradai, and Hossam Afifi, "Game Theoretic Framework for Reputation-based Distributed Intrusion Detection", PASSAT 2013, Washington DC, USA
4. Amira Bradai, Hossam Afifi ,"A Framework Using IBC Achieving Non-Repudiation and Privacy in Vehicular Network", SARSSI 2011, La Rochelle, France
5. Amira Bradai, Walid Ben-Ameur, and Hossam Afifi, "Byzantine Robust Trust Management for Computing ",CollaborateCom 2013, Austin, Texas.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Contents

Trust and reputation management is increasingly attracting the attention of security experts. It is an important way that is used in improving the security in critical systems. The notions of trust and reputation originate from the social sciences which study the dynamics of human societies. The trust concept has been studied in different disciplines from economics to psychology, from sociology to information and computer science.

1.1 Definitions

1.1.1 Trust

Trust is an important metric in interactions which facilitates the formation of functional communities. The measurement of trust has the following characteristics:

- Trust is useful only in an uncertain environment and where the participants need to cooperate with others to achieve their goals. In a predictable and observable environment, trust would be of no significance because each participant would already know the action that could be taken. In addition, if all tasks can be performed satisfactorily by individuals without the need for interactions, it will not be necessary to measure trustworthiness.

- Trust is context-sensitive. It is commonly agreed in the trust literature that trusting is a three-part relationship which can be expressed as "Alice trusts

Bob to do X". The "to do X" part makes a limit on the trust relationship based on how well the subject follows the context in which the relationship exists. For example, Alice may trust Bob to correct her manuscript but she may not trust Bob to solve a mathematic problem. In this case, even if Bob has good intentions towards Alice, benevolence alone does not warrant a high level of trust across all the contexts.

- Trust is subjective. The formation of an opinion about someone's trustworthiness depends not only on the behaviors of the subject but also on how these behaviors are perceived by the agent. The perception of the subject's behaviors often depends on how trustful the agent is towards others and the expectation the agent has on the subject's performance. For instance, although Bob's performance to review is consistent across all customers, Malory, being a more demanding customer than Alice, may trust Bob less than Alice in terms of letting Bob repair his computer.

- Trust is unidirectional. An agent's trust in a subject is based on the knowledge that it has about. This knowledge may be acquired either through the agent's own observations, the recommendations from the agent's friends, or other means. The subject may not necessarily know the agent and therefore may not trust the agent in this case. Even if the agent has direct observations of the subject's past behaviors, the perception of the subject on the agent's performance and benevolence may differ. Thus, an agent's trust in a subject may not be reciprocated.

- Trust may not be transitive. The first time that Alice meets Bob, she does not know to what degree to trust him. Sam, whom Alice trusts, comes forward and vouches for Bob. In this scenario, should Alice trust Bob? The answer can be both yes and no. This depends on the context in which Alice trusts Sam. Sam's opinion of Bob is only useful to Alice if Alice trusts Sam's trust assessment of others.

1.1.2 Reputation

Reputation is public knowledge and represents the collective opinion of members of a community. Reputation is based on the aggregated trust opinion of a group of agents. Since trust is highly subjective, this aggregated result may not be of equal use to all agents.

The difference between trust and reputation can be illustrated by the following statements:

(1) "I trust you because of your good reputation."

(2) "I trust you despite your bad reputation."

Statement (1) reflects that the relying party is aware of the trustee's reputation, and bases his trust on that. Statement (2) reflects that the relying party has some knowledge about the trustee, e.g. through direct relationship, and this factor overrules any reputation that a person might have. This observation reflects that trust is

ultimately a personal and subjective phenomenon that is based on various factors or evidences. Personal experience typically carries more weight than second hand trust referrals or reputation, but in the absence of personal experience, trust often has to be based on referrals from others.

We believe also that reputation is a good measure of someone's contribution to common network operations. Indeed, reputation is usually defined as the amount of trust inspired by a particular member of a community in a specific setting or domain of interest. Members that have a good reputation, because they helpfully contribute to the life community, can use the resources; while members with a bad reputation, because they refused to cooperate, are gradually excluded from the community. So, reputation systems can be called collaborative sanctioning systems to reflect their collaborative nature, and are related to collaborative filtering systems. These systems are already being used in successful commercial online applications

1.2 Motivations

It is important to elaborate models to represent and update trust for different open systems. Any ways to manage information about users in the system represent a significant trend in decision support. But, this management of trust and reputation can face attacks even with central trusted entity. Researches focuses on how to ameliorate the defense mechanisms of reputation and trust models. Different defense mechanisms are adopted for each attack and system. Trust is used as we have previously mentioned for many open systems.

The most known use cases for reputation and trust is the online system. Peer to peer file sharing is one of the applications in which research on reputation and trust is investigated. The challenge behind that is how to select trusted peers for files sharing.

An other motivation for our work is that the majority of systems degrade when maximum verification is applied progressively. Hence, to minimize the consumption of resources due to monitoring, an appropriate optimization for efficient detection is needed for such cooperation.

The last motivation is about distributed reputation and trust systems that are the subject of many issues. The questions are:

• will models effectively aggregate dispersed information and thus weed out incorrect reputation?

• whether media sources, prominent agents, politicians and the state will be able to manipulate reputation and spread misinformation in a network?

To solve this, it is important to study stubborn and forceful peers that can modify the reputation.

1.3 Contributions

The main contributions of these thesis are summarized below:

Contribution 1 :

In the first contribution, we will begin with a general proposed trust model. Then, we don't use the trust directly to identify malicious peers but to feed three centralized reputation models. The first model uses directly trust values to compute a trust vector. The other models consider the credibility to refine the estimated reputation. After the study of some attack on reputation and trust systems, we will classify those cheating behaviors following the Byzantine, Altruistic and Rational (BAR) model. Our proposal is evaluated via simulations with exhaustive potential attack scenarios following our classification. We will show that our refined algorithm is resistant to all different behaviors compared to known proposals as the Eigentrust and methods used in the cloud computing field.

Contribution 2 :

In the second contribution, we want to prove the efficiency of our proposed reputation-based trust model. In the first part of the contribution, we will add refinement to the last efficient model of the first contribution. Globally, the refinement consists in the idea of pre-trusted peers and the privilege peers. To make our idea more clear, we apply it to the job allocation in volunteers computing (cloud computing) and file sharing in peer to peer. We will simulate our new proposed models and we will show the efficiency of new ideas in terms of reducing the job loss, security (safety) of the jobs execution and the file sharing. Generally speaking, we believe that although our mathematical models are tailored for a cloud context, they can be generalized to other distributed peer to peer applications.

Contribution 3 :

In the third contribution, we will focus on the optimization of the reputation and trust system: the optimization of controlling peers behaviors. For that, in general distributed intrusion detection system, we will apply the game theory aspects to save the resource used in controlling. This contribution presents an efficient platform with leader election and decision management based on reputation and game theory.

Contribution 4 :

Unlike the first contribution which will be based on the trust entity, this fourth contribution will be focusing on distributed secure gossip-based reputation management. We will study and validate the convergence of our proposal in two cases : the normal case and the case with stubborn peers. We will implement the model to validate the mathematic demonstration in the two cases. We will analyze the spread of misinformation and persistent disagreement in our model when there is stubborn peers. To study more and more this distributed reputation management, we will test different strategies that can be used by forceful peers in our system. We show the performance of strategies under different types of graphs, scale free, random etc .. We will affirm that forceful peers can manipulate the opinions and affect the final result.

1.4 Organization of the thesis

This thesis is organized into the following eight chapters:

Chapter 2 will represent the state of art of trust and reputation systems. It discusses defense mechanisms against attacks on reputation and trust systems.

Chapter 3 will start with a general reputation model. Then, we will describe how a central entity collects the reputation and executes the trust proposed models. We will propose also our classification of possible cheating problems and we will evaluate our proposal under them. We will show that credibility and trust are efficient to make defense mechanisms.

Chapter 4 will propose a refinement of the models in the third chapter: we will add the pre-trusted peers. Such ideas are applied in the context of hybrid cloud computing and peer to peer file sharing. Hypotheses are verified through simulations to prove the efficiency of such models. Intensives simulations of the job allocation are showing the benefit in terms of job loss. In addition, we will investigate in the context of file sharing in peer to peer. We will affirm the benefit of trust and credibility in minimizing the inauthentic downloads.

Chapter 5 will discuss the optimization problem in reputation management for distributed intrusion detection systems. We will use a simple game theory to solve this problem of monitoring.

Chapter 6 will discuss the aggregation of reputation values from all nodes in the network. A gossip based model is used and dealt with different cases. Simulation results and examples will be presented to validate the convergence of the proposed algorithm. We will discuss also the case of stubborn peers and the persistence of created disagreement.

Chapter 7 will propose a complete analysis of the proposed strategies used by two forceful peers. We will show the performance and the impact in different graphs: scale free, random graph and geometric graph.

Chapter 8 will conclude the manuscript about the contributions of the thesis and will propose some future directions.

Chapter 2

Reputation and trust management

Contents

2.1 Introduction

Trust can be seen as the general confidence in a person or a thing. Generally, it is evaluated by values on a scale from zero to one. In [52], authors explained that there are four major elements structuring trust management:

- initial trust can find its root in social aspects. Marsh, defining trust as a social phenomena, was one of the first authors to introduce a computational model for trust [81].
- trust metrics can be a binary state to express trust and distrust (0 and 1 or positive and negative), opinions or probability metrics. They can be local (where

the evaluation is between two specific users in the trust network) and global (the evaluation is from all the users in the trust network).

- trust propagation is about how it is managed when transitivity is concerned, two operations exist : concatenation and aggregation.

- trust management architecture: reputation systems can be designed in a centralized way or in a decentralized way or a mixture of the two.

  ∗ Centralized system:

  Ratings are reported to a central manager that calculates the reputation of each member as shown in Fig. 2.1.

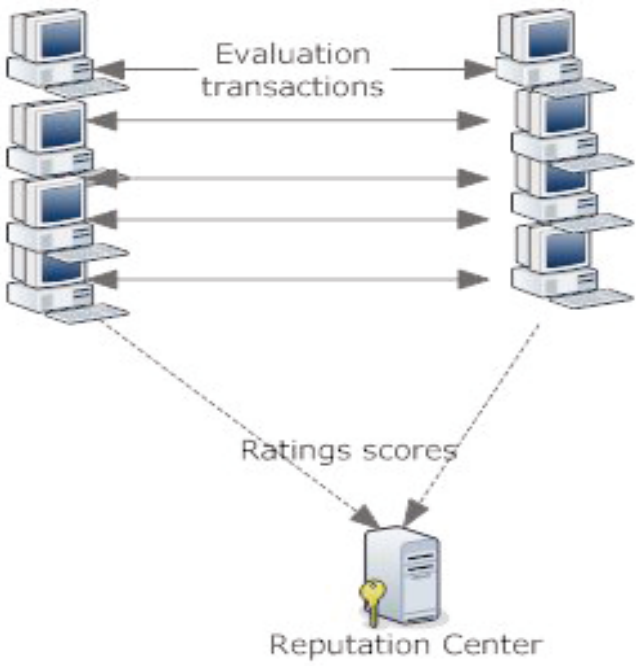

Figure 2.1: General framework for a centralized reputation system

  ∗ Decentralized system:

  In distributed reputation system, several decentralized managers can be at the nodes level. As for recommendations, members can exchange their scores. The calculation of reputation is performed at each member level, depending on ratings received from other members and the personal experience via evaluation transactions as shown in Fig 2.2.

In trust systems, nodes can rely on their own experience on a given node to assess reliability. In addition to personal experience, a node can use the experiences from other peers (indirect experience) obtained via recommendations. Recommendation systems are hence important (social networks, e-commerce, ...etc.). They seek to

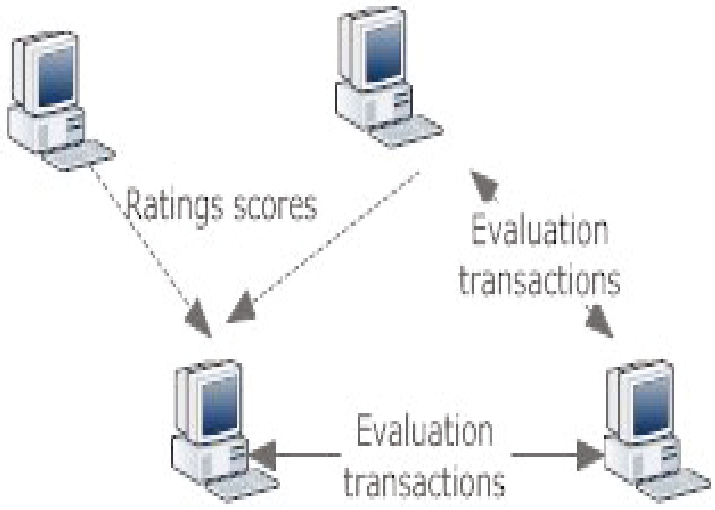

Figure 2.2: General framework for a distributed reputation system

predict the rating or preference that users would give to an item they had not yet considered [50]. This system helps in the evaluation and the propagation of trust in various networks using trust management.

Reputation is a concept closely related to trust. It is what is generally said or believed about a thing. Reputation is seen as one measurable means by which trust can be built, since an entity can trust or distrust another based on good or bad past experience and observation as well as collected referral information about its past behavior.

2.2 Reputation and trust system

2.2.1 Reputation system model

Generally, the collection and the feedbacks formulation is the first step in every reputation system. After the analysis, we come to the aggregation and the dissemination of results. In this subsection, we will explain those three dimensions as being fundamental to any system (shown also in Fig 2.3).

2.2.1.1 Feedback formulation

The individual object collects evidence about other individuals. The first type of evidence is the direct opinion established using the experience of the individual object. The second type is based on the experts' opinions which constitutes a reliable source of information. The third type can be called users' feedbacks . This type is the most popular in recent systems. Usually, human feedback is not reliable because it is the source of attacks. The formulation is a critical component since any weakness in the design of the formulation allows malicious manipulation of reputation.

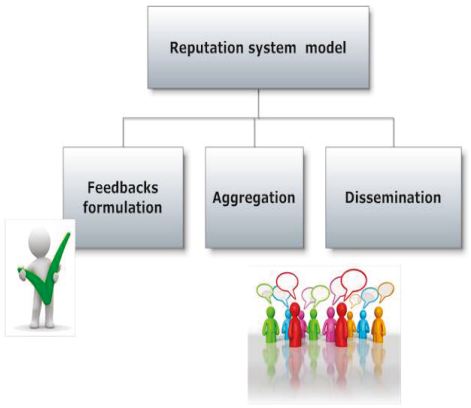

Figure 2.3: Reputation system model

2.2.1.2 Reputation aggregation

Aggregation is a central point in any model of reputation. An aggregation procedure is an operator. It starts from a set of evaluations regarding the same target in the same role to build a new evaluation. So, based on the first step (feedbacks formulation), aggregation mechanisms calculate the reputation scores.

Many models proposed their own solutions for aggregation process. The challenge faced in aggregation models is to be able to describe the true quality of objects even with dishonest ones.

2.2.1.3 Reputation dissemination

All dissemination techniques treat the same problem: some information provided by a single entity which is needed by many others. Since the information can not be passed from the providers to the requesters directly for security reason, a dissemination is needed to provide the requesters with the information.

So, once reputation has been aggregated, it needs to be public and available to requester but not altered. Systems can include extra information to understand the meaning of values (Amazon, Youtube..)

2.2.2 Attacks in trust and reputation systems

2.2.2.1 Self promoting attacks

Attackers manipulate their reputation. They may increase or decrease it in the feedback formulation step. In such self promoting attack, malicious nodes can attack the calculation and the dissemination. We can recall for example the compromise of the logins and the passwords in Ebay system. Here, the dissemination is 100% compromised.

Defense Techniques to mitigate self-promoting attacks include requiring reputation systems to provide accountability, proof of successful transactions, and the ability to limit or prevent an attacker from obtaining multiple identities [9]. The computation dimension should also include mechanisms to prevent colluding adversaries from subverting the computation and storage of the reputation values.

2.2.2.2 White washing

The identity is an important parameter in reputation and trust systems. Using system vulnerability in identity management, whitewashing attack can be launched. For example, an attacker having low reputation can reenter the system with a new identity and a fresh reputation [9].

Defense In the Amazon system, the identity is the name of the product and cannot be changed using simple process. In some systems, the restriction of the number of online pseudonym may be a solution. They use a binding between the user and the IP address [10] and some fees to the new entry. Other systems assign low initial reputation for a new user [11] and initial reasonably reputation based on behavior [12].

2.2.2.3 Dishonest feedback

Many attackers prefer to insert dishonest feedbacks to gain more advantages. Some attackers want to boost some objects by giving good feedbacks to them even if they are not trustful.

Defense

- Some credentials can be asked for providing feedback. Such credentials are a record of transaction, eBay, Amazon [13].
- Mitigating the effect of dishonest feedback: That can be achieved by measuring users' reliability in providing convenient feedback. Methods to compare the feedback reputation of a user have been achieved in the literature. For example, in [14], the weight of the user's feedback is the inverse of the variance in all of this user's feedbacks. In [15], authors proposed a novel personalized approach for effectively handling unfair ratings in an enhanced centralized reputation system.

- Detection of dishonest feedback: a simple way is to detect dishonest feedbacks based on their dissimilarity value from the complete feedbacks set. In some approach like [16], beta probability density functions is used to combine feedback and derive reputation ratings. Based on this, a user can be determined as a malicious user. In [21], a novel entropy-based method is proposed to measure the testimony quality, based on which unfair testimonies are further filtered. The proposed method does not require the assumption regarding the rating distribution.

2.2.2.4 More complex attacks

A malicious peer can be strategic and can build a good reputation and then start cheating occasionally at a variable rate, but still allows other peers to maintain an acceptable reputation. Or, they could oscillate between periods of building and then milking their reputation. To avoid being detected, a malicious attacker can prepare more complicated attacks, that can be grouped:

- In some attack [19], the attacker can provide dishonest feedbacks to some objects at a given time or improve his feedback reputation by being honest. The behavior is dynamically changed all over the time.

- RepTrap attack [20] is a novel attack on feedback-based reputation systems. Attackers find some high quality objects that have a small number of feedbacks. Then, attackers provide a large number of negative feedbacks to these objects such that these objects are marked as low-quality by the system. That is, the system makes wrong judgement about the object quality. As a consequence, the system thinks the attackers' negative feedbacks agree with the object quality and the feedbacks from honest users disagree with the object quality. Therefore, the feedback reputation of the attackers will be increased while the feedback reputation of honest users will be reduced.

Defense To avoid this attack, some models have been proposed [21] [22]. A temporal analysis is proposed such as the changing trend of rating values, the time when feedbacks are provided, etc.. Another analysis is the correlation to find the malicious user. These models are tested with real data. In [22], authors combine the temporal analysis and the user correlation analysis.

2.3 Analysis of existing reputation systems in peer-to-peer and their defense mechanisms

Recently, studies have focused on reputation mechanisms and trust systems specific for P2P like applications [60] (PeerTrust), [113] (EigenTrust) and others for social networks like [59](Semantic Web). In eBay's reputation system, buyers and sellers can rate each other after each transaction, and the overall reputation of a participant

Table 2.1: Comparison of trust system

Evaluation	Trust system
Bayesian network	BNTBM ([28])
Fuzzy logic	PATROL-F([30])
Biology	AntRep([26]) TACS([27]) TDTM([29])
Analytic expressions	GlobalTrust [31] Semantic Web [59]

is the sum of these ratings over the last 6 months. Several trust management protocols [78], [79] and [80] have been proposed for network security, data integrity, and secure routing in different fields. In [80], a group-based trust management scheme for clustered Wireless Sensor Networks (WSN) was proposed. This protocol reduces the use of memory for storing trust scores and minimizes the cost associated with trust evaluation of distant nodes compared to other works. Fuzzy logic was introduced to trust models in the context of sensor networks [83] focusing on the trustworthiness of sensor nodes. It was used to send trusted data between sources and destinations but didn't consider the overhead due to trust in sensor networks.
In this section, we present and describe a set of existing reputation systems and defense schemes.

The aim of reputation-based trust models in peer-to-peer is to identify the trusted nodes among all the nodes. Overall, these models use similar process for the selection of nodes.

The first step is to rely on their own experience with a given peer to assess its reliability (local trust). This trust can be directly assessed by taking into account several factors such as: the quality of previous interactions, their number and the satisfaction obtained after each interaction. In addition to peer's experience, a peer can use the experiences of other peers (indirect experience) to assess the degree of reliability.

The credit for indirect experiments is different from the personal experience of a peer. Models like Semantic Web [59] and Global Trust [31] make the difference between the trust made as a service provider and as recommender (indirect experience) and thus allows to filter malicious peers (peers who recommend malicious peers).

The second step is the calculation of the reputation based on different collected experiences (direct and indirect). At the end of this step, a peer can decide whether to transact with a peer. At the end of each transaction, an other evaluation is made based on the result of the transaction and the service provider is assessed according to the customer satisfaction (reward or punishment).

Reputation and trust evaluation for each model is show in Table 2.1. Many models like AntRep [26] and TACS [27] use the biology. Others like BNTBM [28] use bayesian network. The logic fuzzy allow to model the reputation and trust in simple ways. However, it is very difficult to implement on real networks (scalability problem). Bayesian techniques use the past in considering future exchanges and the future

reputation. They provide flexibility to model different aspects of the trust and context of the environment. But this technique costs time. Wang and Vassileva [17] propose a Bayesian network-based trust model that uses reputation built on recommendations. They differentiate between two types of trust: trust in the host's capability to provide the service and trust in the host's reliability in providing recommendations.

Others techniques use the analytical expressions. Those techniques, used in GlobalTrust and Semantic Web, are easy to read and understand, but do not provide opportunity to model different factors of the trust.

Biology technique have demonstrated their efficiency in scalability problem and application in the peer-to-peer networks. However, in some cases if their approximations lead to select malicious peers as trusted. In the following, we will describe and detail some known reputation and trust systems.

2.3.1 CuboidTrust

CuboidTrust [62], is a reputation-based trust model for global peer-to-peer based on reputation, which is built around four relations based on three trust factors: the peer' contribution in the system (Resources), peer' trustworthiness (calculated from feedbacks) and resources quality. Each of these factors is represented by a vector form of a cuboid coordinates x, y, z and denoted by $P_{x,y,z}$, where z represents the stored resource by y and evaluated by x and $P_{x,y,z}$ the evaluation of the resource z by x. The values attributed to the quality of the resources are binary: +1 (positive) or -1 (negative). The resource is evaluated positively if at the end of a good download (without incidents) the resource is considered honest and authentic. It is evaluated negatively if it does not fit or if the download was interrupted.

2.3.2 EigenTrust

The EigenTrust model [113] computes a global trust metric using system-wide information. This algorithm again focuses on a Gnutella like P2P file sharing network. They based their approach on the notion of transitive trust and addressed the collusion problem by assuming there are peers in the network that can be pre-trusted. While the algorithm showed promising results against a variety of threat models, we argue that the pre-trusted peers may not be available in all cases and a more general approach is needed. Another shortcoming of their approach is that the implementation of the algorithm is very complex and requires strong synchronization of peers. In this model, while the algorithm showed promising results against a variety of threat models, we argue that the pre-trusted peers may not be available in all cases and a more general approach is needed. Another shortcoming of their approach is that the implementation of the algorithm is very complex and requires strong synchronization of peers.

2.3.3 AntRep

AntRep [26] is a distributed reputation model for peer-to-peer. It is based on the paradigm of swarm intelligence and on a system of ants scouts for the construction of

trust relations in the P2P efficiently. In the AntRep model, each peer has a reputation table (RT), like the routing table distance vector. Both tables are different in two different points:

- each peer in the table corresponds to a single reputation.
- the metric is the probability of choosing each neighbor as next hop.

There are two shipping methods for a given ant (with a reputation) peer:

- ants sent to the neighbor with the highest probability reputation in the table,
- ants broadcast to others when there is no preference between neighbors.

This model is selected when no pair path was found or information on this pair are outdated. Once the ants sent the necessary evidence (information on reputation), they are a step backwards (back of ants is generated). Each rollback ants send to a node i reputation table this node is updated at the same time.

2.3.3.1 Semantic Web

Semantic Web [59] is a model where the trust between two peers: P and Q is calculated by adding the trust of interacted agents: in the beginning, a research is done on all paths that connect P and Q, then, for each path, notes associated with each pair on both ends are multiplied. At the end, all scores are added. This model makes trust in two dimensions: trust rating and reliable factor. In this work, authors analyze the similarities between agents in order to make more accurate recommendations.

2.3.4 PeerTrust

PeerTrust [60], is a trust model based on reputation. Authors calculate the credibility of peers based on the feedbacks after each interactions and use trustworthiness factors : the feedback received by other peers, the number of transactions and the credibility of sources. The model has two main features:

- it introduces three trust parameters: the feedback received by other peers, the number of transactions and credibility of peer sources and two adaptive factors the context of the transaction and the context of community (environment) in computing trustworthiness of peers.
- it defines a metric combining all factors to compute the trust.

  We are thinking that in PeerTrust, recommendations are saved distributively searchable by the trustee's id. The mediator does nothing but repeating the set of signed recommendations it is responsible of storing, and can omit information.

2.3.5 XRep

XRep [32] is a P2P protocol where servants can keep track of information about the reputation of other peers and share them with others. Their focus is to provide a protocol complementing existing P2P protocols, as demonstrated on top of Gnutella. However, there are no formalized trust metric and no experimental results in the paper validating their approach. The approach adopts a binary rating system and it is based on the Gnutella query broadcasting method using Time To live (TTL) limit.

2.3.6 SecuredTrust

SecuredTrust [33] is proposed to evaluate agents trust in multi agent environments. SecuredTrust can ensure secured communication among agents by effectively detecting strategic behaviors of malicious agents. After giving a comprehensive mathematical definition of the different factors related to computing trust, authors provide a model for combining all these factors to evaluate trust.

2.4 Conclusion

In this chapter, we briefly presented the trust and reputation aspects and possible attacks. The trust model is composed of three principal mechanisms: feedback formulation, reputation aggregation and reputation dissemination. We have demonstrated the complete framework, attack and defense characterizations. Several key reputation systems and their architecture are represented.

This analysis is valuable for future research in that it provides understanding into the implications of design choices. Reputation systems play an ever-increasingly important part in critical systems. Understanding reputation systems and how they can be compared to each other is an important step toward formulating better systems in the future.

In the next chapter, we will model our trust-based reputation management system. We present new ideas to introduce the credibility to face the attacks on reputation systems. We will compare our model with EigenTrust and the AverageTrust.

Chapter 3

Byzantine resistant trust-based reputation management

Contents

3.1 Introduction

Reputation is a major issue in the hybrid environment because participants are usually unknown to each other since they belong to separate administrative domains [48]. Trust and reputation management can play an important role in the hybrid cloud since this environment is an uncertain and a risky. Thus, trust based reputation management is a specific approach to evaluate and control participants in the system.

For controlling, a good idea is to use root trusted entities to assure an efficient and managed reputation and trust evaluation. Amazon and eBay businesses [49], rely on

the broker network's trustworthiness and reliability. In eBay system, a problem can happen when a failure occurs to the main broker.

In this chapter, we develop a dynamic peer to peer trust model. This model aims to detect possible cheating behaviors both in the private and the IaaS public cloud scenarios. For this purpose, we don't use the trust directly to identify malicious peers but to feed four centralized trust models. The first model uses trust values to compute trust vector. The second, third and fourth models consider the credibility to refine the estimated trust. The last model introduces the pre-trusted peers. Our proposal is evaluated via simulations with exhaustive potential attack scenarios. The evaluation considers several attack scenarios such as grouped attack, non-cooperation in trust evaluation and falsification of trust results. Our refined algorithm is resistant to all different behaviors of various kinds of peers. Generally, we believe that although our mathematical models are tailored for a cloud context, they can be generalized to other distributed peer to peer applications.

3.1.1 General overview of cloud computing

Cloud computing proposes efficient methods for service delivery. It has however several security problems among which the trust in the execution platform. In many cases, the customer doesn't know how trustful the remote cloud service can be. According to [44], there are three major cloud service models:

• Infrastructure-as-a-Service (IaaS): it is a cloud computing service offering on demand processing, network and storage. Here the role of service provider is to manage his machines. He can provide controlling in place regarding how machines are created, memory used, time and performance measured for clearing house procedures. Trust has to be given to the provider as a whole.

• Software-as-a-Service (SaaS): it is a software provided in the form of service and not in the form of a computer program. Cloud providers operate application software in the cloud and users access the software from client front ends. Cloud users do not manage the cloud infrastructure and platform where the application runs. The security problem is how to manage the access to applications (establishing controls and policy models and trusting the operator).

• Platform-as-a-Service (PaaS): it is a platform service hosted by an operator and accessed from internet. Cloud providers deliver a computing platform that typically includes operating system, programming language execution environment, databases, and web servers. The primary security problem is on protecting data (storage as a service) and the ability to encrypt the data.

So it is clear that especially in IaaS, trust remains a decisive issue.

Cloud computing is usually deployed in one of three scenarios:

• Private clouds built for the exclusive use of one client, providing the utmost control over data, security, and quality of service. Here trust can be totally granted to the service.

• Public clouds include Amazon Elastic Cloud Compute, Google App Engine [4], run by third parties and applications from different customers, are likely to be mixed together on the cloud's servers, storage systems, and networks.

• Hybrid clouds that combine both public and private cloud models can help to provide on-demand, externally provisioned scale services. Systems like Cloud@Home [46] and Nebulas [47] discuss deploying cloud services in a distributed volunteer way. In fact, deploying cloud services in such systems comes with advantages and drawbacks: Hybrid execution improves usage of available resources. It provides scalability and low cost deployment. But, malicious parties in public cloud may take advantage of the system by deliberately performing poorly and being dishonest or claiming more running time for monetary benefits. We focus on the possibility to use a hybrid cloud (IaaS model shared with private cloud) and still guarantee a good degree of security and prevention of multiple threats. To assure this prevention, we must control reliability of cloud resources. Moreover, the job submission strategy must be wise to choose the proper peers for submitting jobs. A malicious group can also subvert the system and cause disastrous results. These disadvantages can be addressed with a reputation based trust management system, which can effectively filter out poor performing and malicious nodes.

Applications like PSAs (Parameter Survey Applications) need hybrid execution mechanism utilizing both local computing resources with a batch scheduler and an IaaS Cloud. In a cloud security survey [74], 32% of enterprises are studying the opportunity of moving applications in hybrid clouds (10% in production, 21% in implementation and 24% piloting). However, the hybrid cloud is the most critical in terms of identity management, open client, location awareness, metering, management and governance. Cloud computing systems offer infrastructures for applications shared by multiple tenants. These tenants are with different security domains. Moreover, enterprises rely on the security processes and algorithms put in place by providers, while not being able to verify the security of their applications.

3.1.2 Trust and cloud computing

Several studies in research and industry combine trust management and cloud computing environment [63]. Some trust models in grid computing apply trust for enhancement of resource allocation [117] [118]. In [66], authors include an independent trust management module on the top of other security modules. They provide trust strategies respecting cross-clouds environments. In [67], the trust model for cloud resource is based on a security level evaluator (by authentication type), a feedback evaluator and a reputation evaluator. The proposed trust management in cloud computing are based on a simple arithmetic sum like in [67]. Moreover, the models proposed for P2P and distributed network have not been tested in cloud computing environments. In [68], authors present a trust model to solve security issues in cross-clouds environment. The model is composed of three parts: trust decision for the customer and the provider, recommendation and trust update. In [68], the results of the presented work for trust management in cloud are not based on theoretical foundation. In [69] a formal trust model is given for grid computing. The architecture combines trust based authentication, job request and recommendation for trust update. In [69], there are not specialized on how to take efficiently the recommendation.

In [110], we use some algebraic methods to evaluate the trust in multi-domain cloud based applications.

Our model in this chapter computes global trust needed for submitting tasks in public cloud taking into consideration the credibility factor in an efficient way to reduce the false alarms. The rest of the chapter is organized as follows: Section 3.2 exposes the system architecture and the attack model. Section 3.3 describes the proposed reputation model and algorithms for trust management. Section 3.5 shows simulations results and comparison with existing works. Finally, the chapter concludes and lists new ideas that can be fetched, in the section 3.6.

3.2 System overview

In this section, we first describe our network architecture. Then, we describe the possible behaviors of peers in the system following the BAR (Byzantine, Altruistic and Rational) model [71].

3.2.1 System architecture

We start from the assumption that IaaS cloud is used to remotely execute some tasks of an application. Parameter Surveys Applications (PSA), is an example of applications executed in such hybrid environment. This application is composed of many tasks, one part of tasks executed in the IaaS and another part in the local resources of the enterprise (the private cloud). So, let us assume that there are N peers (a finite set of local resources of the enterprise and peers allocated from the IaaS cloud computing) executing tasks for the same application, a portal and a scheduler which are shown in Fig. 3.1.

- The portal represents the interface to execute the application. It brings the other parties together. The portal distinguishes between two types of peers: local and remote.
  The local peers are supposed to be in a trusted and protected location but they can launch internal attacks.
  Remote ones are supposed to be in different domains, that are not necessarily well protected.

- The scheduler organizes the application's execution. It implements the execution plan.

- The user submits the application and fixes the deadlines and execution time with the portal.

Fig. 3.2 gives more details about the architecture of our system. Reputation data that is needed for the trust evaluations of the peers, stored in the portal. It stores also the database of reputations. Its task is to compute the reputation value for each peer and make the decision. For that, the portal has three modules:

- Reputation collector: responsible for retrieving local reputation vectors.

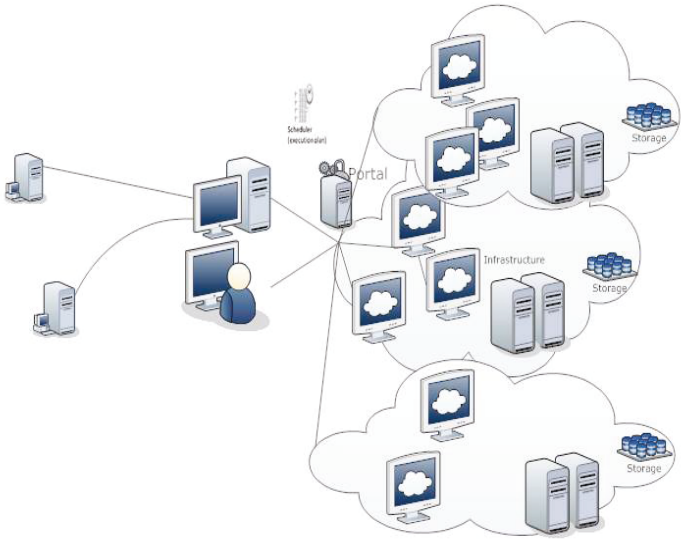

Figure 3.1: System overview

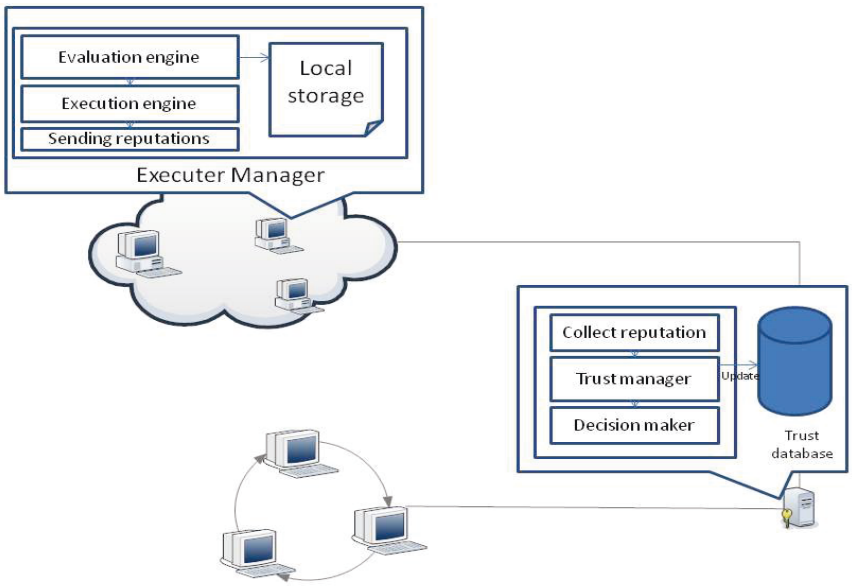

Figure 3.2: Architecture details

- Trust manager: responsible for calculating the global trust vector. We can notice in this module that the trust is performed in a dynamic and centralized way (in a trusted entity).
- Decision maker: responsible for deciding for current and future execution. The decision is based on the final trust vector. It consists on guiding the scheduler in the application organization. This means that if the trust score is good, the scheduler will keep the peer in the execution plan.

In our system, each peer has three engines and a local storage component on its executer manager:

- Evaluation engine: responsible for a cyclic update of evaluations using the model explained in the section 3.3.1.
- Execution engine: responsible for the task execution and results delivery to other peers.
- Reputation sender: responsible for sending evaluations to the portal.

3.2.2 Some possible peers behavior

The executer in our system has two possible behaviors: a normal behavior or a cheating behavior. We can clarify the behaviors by using the BAR model recalled below.

Definition 1 *BAR model [71], Byzantine Altruistic Rational model (known as BAR) is a model of computer security for distributed systems used to serve as an error detection model. Currently it is mainly used in P2P systems. A peer can be classified into one of three categories that represent the BAR model, namely:*

- *Altruistic : the peer is considered as a peer working accurately according to the protocol.*
- *Rational : the peer is only interested in optimizing the use of its resources. It does deviate from the protocol used if it considers that because of using this protocol, its performances decrease.*
- *Byzantine : (or malicious) peer does not follow the used protocol, either because it is compromised (or not well configured) or because it follows a power optimization of resources that is not the same as for rational peer.*

In our system, peers are classified into two types of behaviors:

- Normal behavior : this corresponds to the behavior of altruistic peers.
- Cheating behavior : peers that are rational or byzantine. This type of peer adopts resource optimization strategy or will not respect the reputation management model. They can choose not to cooperate in the evaluation process. The probability to have cheating local peers is less than to have cheating remote ones.

3.3 Proposed trust management system

In this section, we propose a trust model and four reputation management algorithms based on algebraic calculations : PerronTrust (can be called Power trust), CredTrust, CredTrust-trust and CredTrust-trust with pre-trusted peers.

The first approach is basically a power method to compute the final reputation vector. Since the convergence of the algorithm is based on Perron-Frobenius theorem, we call it PerronTrust. To make PerronTrust more resistant to attacks, we add a credibility parameter. So, the trust is calculated based on the credibility of peers. This idea will be illustrated in CredTrust model. To take into account more sophisticated attacks, a further enhancement is proposed leading to CredTrust-trust model. Details will follow in Section 3.3.2.

All these reputation models use the same trust model described in Section 3.3.1. The trust values are maintained in the trusted entity "Portal" that can compute the credibility and the reputation in an efficient and objective way. The advantage is that this entity is a trusted one and can decide for the future of the application and avoid the peer subjectivity problem.

3.3.1 Proposed trust model

Definition 2 *Let T be a trust matrix. In order to have the global view of the trust management, we construct this matrix of peer to peer trust scores. It is initialized to be 0.5 (ignorance).*
The result of the trust evaluation process is this matrix, T containing peer to peer trust values between peers.

Definition 3 *The P2P trust score $T_{ij}(t)$ is the evaluation of peer i to peer j at time t. $T_{ij}(t)$ depends on time.*

Definition 4 *$d_{ij}(t)$ is the direct evaluation of behaviors.*

$d_{ij}(t)$ is updated through verifications like challenge sent periodically: it represents the fraction of positive results when peer i verifies peer j. As we said before, the verification is in term of time taken to do a job, crashes of a node, and correctness of returned results. For example, after verifications between two peers 1 and 2 with 4 positive results and 5 negative results, the evaluation score is 4/(4+5)=0.44. So, $d_{ij}(t) = 0.44$.

Definition 5 *δt is the update period*

The trust value is evaluated following the Eq.3.3.1.

$$\mathrm{T}_{ij}(t) = \begin{cases} p.T_{ij}(t-\delta t) + (1-p).d_{ij}(t), & \text{if i verifies j;} \\ T_{ij}(t-\delta t), & \text{else.} \end{cases} \quad (3.1)$$

In Eq 3.3.1, we want to assign more weight to recent interactions and less weight to the older ones. So, p (used in the equation) is used for that purpose.

3.3.2 Proposed reputation models

Let us first introduce some definitions and notations needed for our models. Then, we describe the proposed algorithms to compute the reputation.

$||x||_1$ and $||x||_2$ respectively stand for the l_1-norm and l_2-norm of vector x.
T^t is the transpose of the matrix T.

Definition 6 *Let π be the reputation vector of size N. Initially, all peers are equally trusted, i.e $\pi_i = 0.5$ where $i = 1, 2, ..., N$. It is represented as a real number in the range of [0, 1] where 1 indicates complete trust, 0.5 ignorance, and 0 distrust.*

Definition 7 *ε is the convergence threshold. It is the value needed to determine when the vector in question is stable and is not changing.*

Definition 8 *Threshold is the reputation limit used by the portal in the "decision maker" module.*

3.3.2.1 PerronTrust model

The PerronTrust algorithm is a power method (method to find the first eigenvector and eigenvalue of a matrix). Notice that the power method is used in different areas including, for example, the computation of the PageRank of web documents [112]. PageRank represents a way of ranking the best search results based on a page's reputation. It ranks a page according to how many other pages are pointing to it. To derive a reputation score, they combine the collection of hyperlinks to a page seen as public information. Google's search engine is based on this PageRank.

Using the trust matrix T and the current trust vector, we get a new approximation of the reputation of each peer j through a combination of the trust values of j:

$$\pi_j \leftarrow \frac{\sum_{i=1}^{N}(T_{ij}.\pi_i)}{||\pi||_1}. \tag{3.2}$$

If a peer i has a high reputation score, then it is natural to give more importance to the trust values that it is assigning to other peers. Writing Eq.3.2 for each peer leads to Eq.3.3:

$$\pi \leftarrow \frac{T^t.\pi}{||\pi||_1}. \tag{3.3}$$

The reputation will then be iteratively computed by repeating Eq.3.3 until the vector becomes stable. The convergence of Alg 1 is based on the Perron-Frobenius theorem that we recall here for sake of completeness. The Perron Frobenius theorem asserts that a real square matrix with strictly positive entries has a unique largest real eigenvalue and that the corresponding eigenvector has strictly positive components.

Theorem 1 *Alg 1 will converge to the final reputation vector result π after a certain number of iterations.*

Algorithm 1: PerronTrust algorithm

Require: ε, N
Ensure: π
1: Retrieve trust values (in T)
2: Initialize the reputation vector π
3: **while** $||\pi(t) - \pi(t-1)||_1 > \varepsilon$ **do**
4: Calculate the reputation vector : $\pi \leftarrow \frac{T^t.\pi}{||\pi||_1}$
5: **end while**

Proof: Since we can assume that T is the real square matrix with positive entries, then we know that by a repetitive application of the previous iterative formula, the vector π will converge to the unique eigenvector associated with the largest eigenvalue $\lambda_{max}(T^t) = \lambda_{max}(T)$. Notice that we will get $||\pi||_1 = \lambda_{max}(T)$. The convergence is guaranteed if the starting point (the first approximated score vector) is not orthogonal to the eigenvector. This is clearly satisfied by the positive vector $\pi = (0.5, 0.5, 0.5...)$ since the eigenvector is also positive by the Perron-Frobenius theorem. ■

The iterative computation in Alg 1 continues until the total difference between $\pi(t)$ and $\pi(t-1)$ becomes smaller than ε. We have proved this convergence just before. The convergence threshold is often predefined by the portal of the application. The threshold can be adapted depending on the precision of the decision wanted by the portal.

3.3.2.2 CredTrust model

In the previous model, some peers might have a good reputation score while they are not really able to give a good estimation of the trust of the other peers. Then, it is important to assign less importance to their evaluation. For that, we want to introduce a parameter that can measure this feature. This parameter is called credibility.

If a peer gives wrong evaluation about other peers, its credibility value is decreased and its evaluation values have a reduced impact on the trust of other peer. Similarly, if a peer's evaluation is good and in agreement with other evaluation peers, its credibility should be high. The credibility of a peer is used to weight the feedback it reports.

Let us first add some definitions:

Definition 9 *Let* ***Cred*** *be the vector containing the ability to evaluate correctly the trust of peers (credibility).* $Cred_i$ *corresponds to the credibility of peer* i*. Credibility values are normalized so that they lie between* 0 *and* 1*.*

Given the trust value of peer j seen by peer i and the reputation estimated for peer j in the portal, we can evaluate the global credibility of the peer i using the Formula 3.4:

$$\text{Cred}_i = 1 - \left(\frac{\sum_{j=1}^{N} |\pi_j - T_{ij}|^2}{\sum_{j=1}^{N} |\pi_j - [1-\pi_j]|^2} \right)^{\alpha} \quad (3.4)$$

where $[1-\pi_j]$ denotes the nearest integer to $(1-\pi_j)$ and α is fixed number. Observe that $[1-\pi_j]$ is equal to 0 if $\pi_j > 0.5$ and to 1 if $\pi_j < 0.5$.

The credibility of i is equal to 0 if the evaluation given by i is always the farthest possible evaluation that can be accepted by the system. Conversely, if T_{ij} is equal to π_{ij} for each j, then $Cred_i$ is equal to 1.

Given these credibilities scores of each peer i in the system, it is now natural to estimate the global reputation vector using the formula 3.5. To have reputation between 0 and 1, we divide by $||Cred||_1$.

$$\pi \leftarrow \frac{T^t.Cred}{||Cred||_1} \quad (3.5)$$

The algorithm CredTrust performed by the portal is summarized below:

Algorithm 2: CredTrust algorithm

Require: ε, N
Ensure: π
1: Collect trust evaluation T
2: Initialize the reputation vector π
3: Initialize the credibility vector Cred
4: **while** $||Cred(t) - Cred(t-1)||_1 > \varepsilon$ **do**
5: **for** $i \in \{1, ..., N\}$ **do**

$$Cred_i = 1 - \left(\frac{\sum_{j=1}^{N} |\pi_j - T_{ij}|^2}{\sum_{j=1}^{N} |\pi_j - [1-\pi_j]|^2} \right)^{\alpha}$$

6: **end for**
7: $\pi \leftarrow \frac{T^t.Cred}{||Cred||_1}$
8: **end while**

All our simulations show that convergence is obtained. Formula 3.5 can be written in the form $\pi \leftarrow f(\pi)$ where f is a function obviously defined by combining formula 3.5 and 3.4. The function f is clearly continuous. Each vector π belonging to the convex hull of the rows of the matrix T (columns of T^t) is mapped to a vector $f(\pi)$ belonging to the same convex hull. Using Brouwer fixed-point theorem, we can deduce that f has at least one fixed point: i.e., a vector π such that $f(\pi) = \pi$. A deeper study of the function f is required to deduce that the iterative process $\pi \leftarrow f(\pi)$ converges to a fixed point.

3.3.2.3 CredTrust-Trust ($CredTrust^2$) model

As it will be shown in the simulation section, while the CredTrust approach is generally more efficient than the PerronTrust approach, there are some situations where it

gives less precise results. This happens if there are some malicious peers who decide to correctly evaluate most of the other peers except one chosen malicious peer who is intentionally given a good evaluation. Since the malicious peers are giving the right evaluation in almost all cases, they will have a high credibility. This means that the evaluation given by these malicious peers will have more impact on the final result. Since these malicious peers decided to over-estimate a chosen malicious peer, this malicious peer will have a final reputation higher than the one obtained by PerronTrust approach.

To overcome this problem, we are going to re-introduce again the trust in the iterative process. Instead of using only the credibility (as in CredTrust) or the trust (as in PerronTrust), we combine both of them.

More precisely, we modify the Eq. 3.5 to have the new following formula :

$$\pi \leftarrow \frac{T^t.(Cred \otimes \pi)}{||Cred \otimes \pi||_1} \tag{3.6}$$

where $Cred \otimes \pi$ denotes the componentwise product of $Cred$ and π. It is now clear that even if a peers has a high credibility, the impact of its opinion is attenuated if he has a low trust.

The same convergence remarks related to the previous algorithm apply also for Alg 3.

Algorithm 3: CredTrust-Trust algorithm

Require: ε, N
Ensure: π
1: Collect trust evaluation (in T)
2: Initialize the reputation vector π
3: Initialize the credibility vector Cred
4: **while** $||Cred(t) - Cred(t-1)||_1 > \varepsilon$ **do**
5: **for** $i \in 1..N$ **do**

$$Cred_i = 1 - \left(\frac{\sum_{j=1}^{N} |\pi_j - T_{ij}|^2}{\sum_{j=1}^{N} |\pi_j - [1 - \pi_j]|^2} \right)^{\alpha}$$

6: **end for**
7: $\pi \leftarrow \frac{T^t.(Cred \otimes \pi)}{||Cred \otimes \pi||_1}$
8: **end while**

3.3.2.4 Pre-trusted peers for CredTrust-trust

In some cases, there are some pre-trusted peers that are known to be trustworthy in any system.

The first few peers to join the system are generally known to be pre-trusted peers, because these peers hardly have any motivation to destroy the system. We assume for example that the portal has a knowledge of some trusted peers.

The portal maintains unchanged, the reputation scores for peers that it trusts, in each step. With some modifications in Alg 3, we can benefit from the trusted peers by fixing the initial scores. Also, we affect at the end the modified trust score according to the knowledge that the peer has.

Definition 10 *If some set of peers, denoted by PTP (Pre-Trusted Peers), among all M peers are known to be trustworthy, a trust value PTV is assign.*

Hence, the initial trust score is presented as a vector h: h=(0.5,0.5,0.5,PTV,0.5,PTV,0.5) as an example.

Algorithm 4: CredTrust-Trust algorithm with pre-trusted peers

Require: ε, N
Ensure: π
1: Initialize the reputation vector h: h=(0.5,0.5,0.5,PTV,0.5,PTV,0.5)
2: **while** $||Cred(t) - Cred(t-1)||_1 > \varepsilon$ **do**
3: **for** $i \in \{1, ..., N\}$ **do**

$$Cred_i = 1 - \left(\frac{\sum_{j=1}^{N} |h_j - T_{ij}|^2}{\sum_{j=1}^{N} |h_j - [1-h_j]|^2} \right)^{\alpha}$$

Affect the pre-trusted peers with the corresponding trust score PTV
4: **end for**
5: $h \leftarrow \frac{T^t.h}{||h||_1}$
6: **end while**

In the algorithm 4, we presented the execution of the pre-trusted method using the PTV=0.9. The iterative computation in continues until the total difference between $Cred(t)$ and $Cred(t-1)$ becomes smaller than ε.

3.4 Simulation analysis

In this section, we analyze the performance results of the three proposed approaches described in the previous section. We describe a set of attacks and behaviors that are considered here. Then, we compare our three models PerronTrust, CredTrust and CredTrust-trust to AverageTrust and EigenTrust under illustrated attacks.

3.4.1 Simulation parameters

We first consider a hybrid execution with 20 peers in an IaaS cloud and 80 peers running on local resources. So, the total number of peers in the system is $N = 100$. Peers can be either normal or cheating. The behavior of each peer is chosen randomly depending on:

1/ Whether the peers is a local one or it belongs to the IaaS,
2/ The attack scenario (peers behavior) described below.

In CredTrust and CredTrust-trust, a coefficient α is needed to compute credibilities. To analyze the effect of this coefficient, we vary it and observe its effect on the trust and reputation management. Fig.3.3 shows that the number of iteration (or cycles) decreases when α increases. However, if α is very large, the credibilities of all peers is very close to 1 which is equivalent to compute π as the barycenter of the N rows of T: all peers have the same weight here. In other words, by considering large values of α we will loose the strength of the credibility idea.

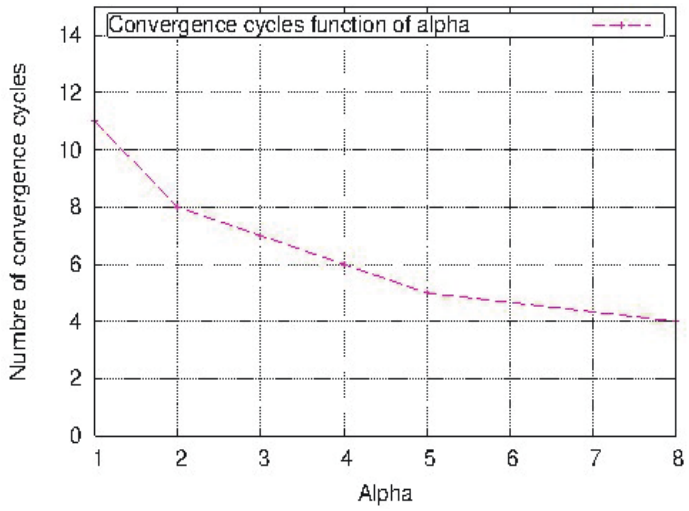

Figure 3.3: Configuration of α

It is also observed that both algorithms CredTrust and CredTrust-trust converge for any value α. Moreover, changing α does not affect the trust vector in very significant way. So, we decided to take $\alpha = 3$ in our simulations.

3.4.2 Cheating peers' behaviors

In this section, we present some of the possible behaviors of the peers that are considered in our experiments.

3.4.2.1 Byzantine peers' behaviors

As we explained before in the BAR model, byzantine peers do not follow the proposed model to manage the reputation. We can list some possible behaviors in this case:

- Overestimating peers : this type of peers always overestimates the scores of peers with which they interact. They can overestimate the byzantine peers.
- Inverting peers : this type of peers always inverses the scores of reputation obtained after interaction with others (gives a bad evaluation to honest peers and a good evaluation to malicious peers).

- Coalition peers : peers form a malicious group that assign a high reputation value to most of the other peers except one malicious peer who is intentionally given a very good evaluation.

3.4.2.2 Rational peers' behaviors

As we explained before in the BAR model, rational peers deviate from the proposed model to optimize resources or due to the configuration. We can list some possible behaviors in this case:

- Non cooperative peers : this type of peers manifests itself as being uncooperative. For saving resources, this peer may not participate in the evaluation process.
- Less efficient peers : partially altruistic peers are rational, they want to optimize resources of interaction with the evaluated peer. They are less efficient and affect the computation of the reputation.

3.4.3 Results and comparison analysis

This section focuses on the simulation of the different attack scenarios described before and try to analyze the results. Comparison of the results of our scheme with the EigenTrust [113] scheme and AverageTrust under these scenarios is done. We compare also the final trust assign to cheating peers and the difference between normal peer and cheating peer of our three schemes with AverageTrust scheme and EigenTrust. The difference between normal peer and cheating peer, is the ratio of trust assign to cheating peers over trust assigned to normal peers, is used to evaluate the accuracy of the decision based on trust.

Notice that to compare our work to EigenTrust, we consider only the ratio of trust assigned to cheating peers over trust assign to normal peers. This choice is due to the convergence of EigenTrust to collinear vectors for different simulations. So in each convergence (simulation) the trust of cheating peer change but the ration is the same. That is why we consider the stable metric. Before the results of the comparison, an overview of EigenTrust and AverageTrust is proposed below.

3.4.3.1 EigenTrust Model

In EigenTrust [113], the global reputation of each peer i is given by the local trust values assigned to peer i by other peers, weighted by the global reputations of the assigning peers. An overview is proposed in Alg 5.

3.4.3.2 AverageTrust Model

This method is used for eBay and some models in cloud computing. An overview is proposed in Alg 6.

Algorithm 5: Basic EigenTrust algorithm

Require: ε, N
Ensure: π
1: Retrieve trust values (in T)
2: Normalize the values in T to have stochastic matrix $T2$
3: Initialize the reputation vector π
4: **while** $||\pi(t) - \pi(t-1)||_1 > \varepsilon$ **do**
5: Calculate the reputation vector : $\pi \leftarrow T2^t.\pi$
6: **end while**

Algorithm 6: Basic AverageTrust algorithm

Require: ε, N
Ensure: π
1: Retrieve trust values (in T)
2: Initialize the reputation vector π
3: Calculate the reputation vector :
4: **for** $i < N$ **do**
$\pi_i \leftarrow Average(T_i)$
5: **end for**

3.4.3.3 Byzantine peers' behaviors: overestimating peers

We consider the case of overestimating peers: Byzantine peers overestimate all peers including the other Byzantine peers. Let us suppose that after interaction, the trust values given by peers in the system are as follow:

- overestimating peers estimate all peers as trusted (=1),
- altruistic peers estimate byzantine peers with 0.2 (or another very bad value) and others with 0.8 (or another very high value).

Results are shown in Fig. 3.4 where we consider the three algorithms CredTrust-trust, PerronTrust and CredTrust compared to AverageTrust, with different densities of malicious peers. The final reputation score of malicious peers is represented for each algorithm.

Observe that when the density of malicious peers increases, the trust level of these peers increases. This is very clear for PerronTrust, CredTrust and averageTrust, while the increase is much slower for CredTrust-trust. With more than 40% of malicious peers, CredTrust-trust shows good results comparing to PerronTrust, CredTrust and AverageTrust. This is due to the fact that CredTrust-trust takes into account both the credibility and the trust level when the reputation vector is updated.

Notice also that even PerronTrust seems to be slightly better than CredTrust for this scenario. In fact, even if the malicious peers are here overestimating all other peers, their credibility is not low since they propose a good evaluation of altruistic peers. This means that the impact of their evaluation is still high when CredTrust is used, while this impact is lower in the case of PerronTrust since the weight of the

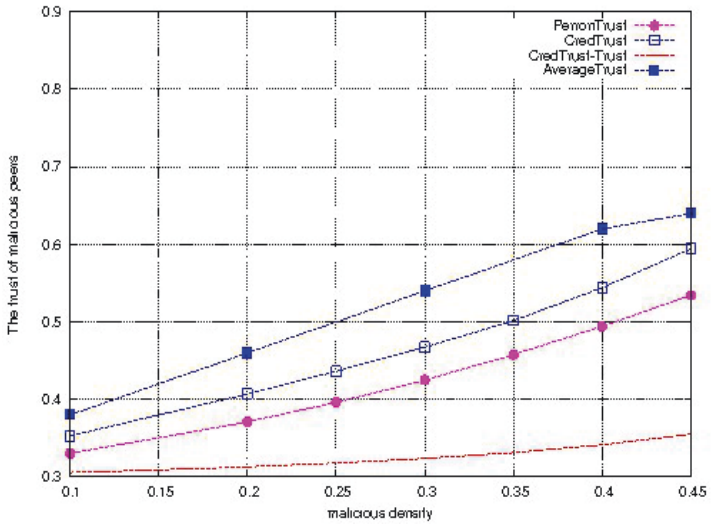

Figure 3.4: Trust under overestimating byzantine peers' behaviors

evaluation given by a peer is equal to his trust. Observe also that the AverageTrust shows high reputation value comparing to our proposed algorithms.

Suppose now that we want to compare our three algorithms to EigenTrust and AverageTrust in term of $RTO(RatioTrustOverestimating) = \frac{Trust\ of\ overestimaing\ peers}{Trust\ of\ altruistic\ peer}$. This parameter is important because it shows how the differentiation between the overestimating peers and altruistic peers will be clear when the ratio is small.

Results are shown in Fig. 3.5 where we consider the three algorithms CredTrust-trust, PerronTrust and CredTrust compared to EigenTrust and AverageTrust, with different densities of malicious peers.

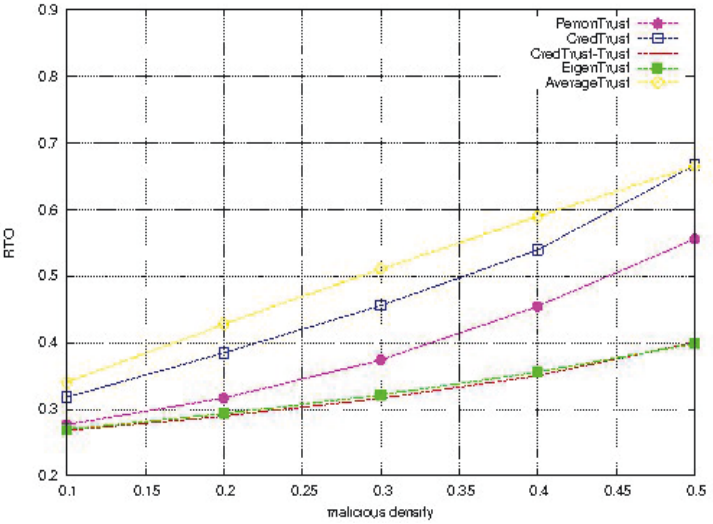

Figure 3.5: RTO under overestimating attack

Observe also that even CredTrust-trust and EigenTrust seems to have slightly same performance in this case. The impact of the evaluation of overestimating peers is lower in CredTrust-trust and EigenTrust than in PerronTrust, CredTrust and AverageTrust. So, reputation of overestimating peers is low in CredTrust-Trust. That is why the ratio RTO is small.

3.4.3.4 Byzantine Peers' Behaviors: inverting peers

We suppose here that all peers cooperate in the evaluation of reputation values. the trust values given by peers in the system are as follow:

- inverting peers estimate inverting peers with 0.8 (or another very high value) and altruistic peers with 0.2 (or another very bad value),
- altruistic peers estimate byzantine peers with 0.2 and others with 0.8.

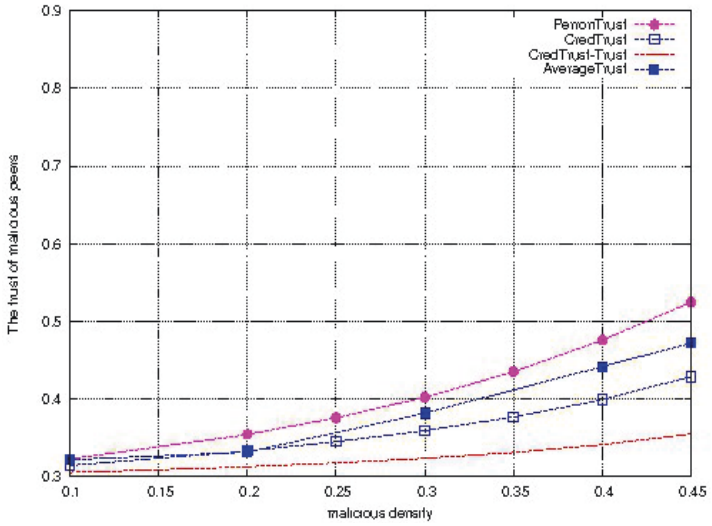

Figure 3.6: Trust under Byzantine peers' behaviors: inverting peers

The reputation values estimated for inverting peers are shown in Fig. 3.6. When the proportion of malicious peers is about 10%, the four algorithms give almost the same estimation for the trust of malicious peers. However, when the fraction of malicious peers increases, the trust values increase for the three models.

Not taking into consideration the credibility factor, PerronTrust and AverageTrust can not punish the peers that give wrong trust values. While CredTrust is doing better than PerronTrust since it is considering the credibility of peers, CredTrust-trust outperforms PerronTrust, CredTrust and AverageTrust. This is again due to the fact that CredTrust-trust penalizes more the inverting byzantine peers.

Suppose that we want to compare our three algorithms to EigenTrust and AverageTrust in term of $RTI(RatioTrustInverting) = \frac{Trust\ of\ inverting\ peers}{Trust\ for\ altruistic\ peers}$

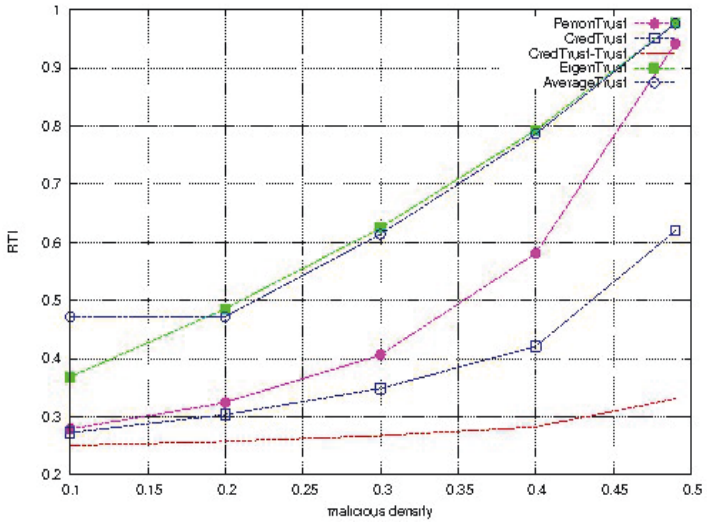

Figure 3.7: RTI under inverting byzantine peers' behaviors

Results are shown in Fig. 3.7 where we consider the three algorithms CredTrust-trust, PerronTrust and CredTrust compared to EigenTrust and AverageTrust, with different densities of malicious peers.

Observe that CredTrust-trust outperforms PerronTrust, CredTrust, EigenTrust and averageTrust. This is again due to the fact that CredTrust-trust penalizes more the inverting byzantine peers. It is clearly that our three methods compared to EigenTrust and AverageTrust help in taking secure decision and then assure trusted execution of tasks in the public cloud.

3.4.3.5 Byzantine peers' behaviors: different level of maliciousness

Consider that we have two groups of Byzantine peers:

- Group 1: Peers inverse the trust of others. They estimate peers from group 1 and 2 with 0.8 and altruistic peers with 0.2,
- Group 2: Peers different from altruistic peers and peers from group 1 (They don't inverse the scores). They estimate peers from group 1 and 2 with 0.6, altruistic peers with 0.2.

We will assume that 20% of the peers belong to group 1, 20% belong to group 2 and 60% of the peers are altruistic.

We consider also the following ratios:

- $R1 = \frac{Trust\ for\ group\ 1\ peers}{Trust\ for\ altruistic\ peers}$
- $R2 = \frac{Trust\ for\ group\ 2\ peers}{Trust\ for\ altruistic\ peers}$

The following obtained conclusions are from results in the table 3.1:

- In this scenario, we affirm that the credibility of peers from group 1 is clearly less important in CredTrust-trust than in CredTrust,

Table 3.1: Results under different levels of maliciousness

	PerronTrust	CredTrust	$CredTrust^2$	Eigen	Average
Reputation for group 1 peers	0.3204	0.3239	0.2364		0.4020
Reputation for group 2 peers	0.3216	0.3265	0.2371		0.4040
Reputation for altruistic peers	0.6833	0.6794	0.7670		0.6020
Credibility for group 1 peers		0.2922	0.2587		
Credibility for group 2 peers		0.5844	0.5173		
Credibility for altruistic peers		0.8164	0.9523		
R1	0.468	0.476	0.308	1.002	0.667
R2	0.456	0.480	0.309	1	0.671

- the credibility of peers from group 2 is clearly less important in CredTrust-trust than in CredTrust,
- the credibility of peers in group 1 is less important than the credibility of peers in group 2,
- the trust assigned to peers from group 1 is clearly less important in CredTrust-trust than in PerronTrust and CredTrust and AverageTrust,
- the reputation assigned to peers from group 2 is clearly less important in CredTrust-trust than in PerronTrust , CredTrust and AverageTrust,
- altruistic peers are more recognized in the CredTrust-trust model than in PerronTrust, CredTrust and AverageTrust,
- R1 is less important in CredTrust-Trust than in AverageTrust, CredTrust and PerronTrust,
- R2 is less important in CredTrust-Trust than in EigenTrust, AverageTrust, CredTrust and PerronTrust,
- EigenTrust can't detect the byzantine peer in this case since ratios R1 and R2 are equal to 1.

In this case also, CredTrust-trust ($CredTrust^2$) outperforms PerronTrust, CredTrust, AverageTrust and EigenTrust.

3.4.3.6 Byzantine peers' behaviors: coalition

We consider that malicious peers select one attacker of the coalition. They assign to this attacker good trust (= 1). Malicious collusive peers provide true trust to hide their essences. The attacker evaluates peers in a consistent way (like an altruistic peer) in order to increase his credibility.

We will focus on the reputation value of the attacker chosen by the coalition. We still use the three algorithms, EigenTrust and AverageTrust to compute the reputation vectors.

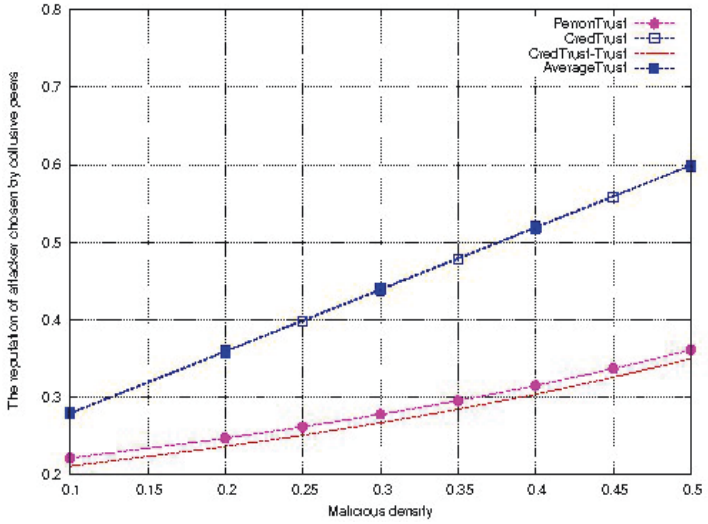

Figure 3.8: Trust under coalition

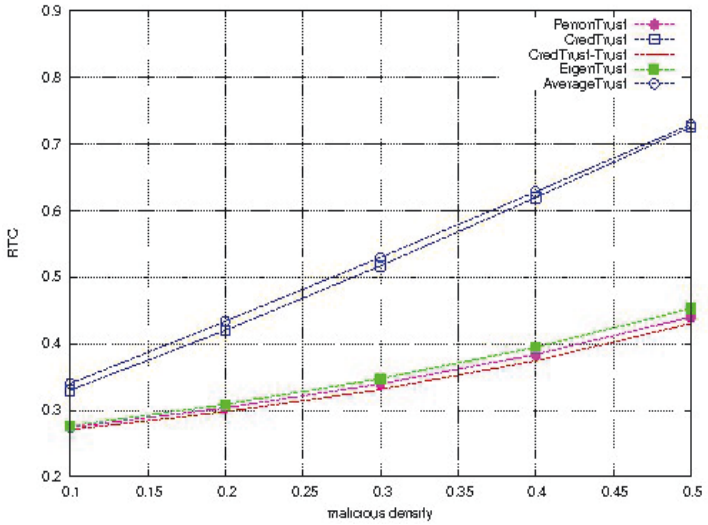

Figure 3.9: RTO under coalition

Table 3.2: Results under non cooperative peers

	PerronTrust	CredTrust	$CredTrust^2$	Eigen	Average
Reputation for inverting peers	0.3286	0.3256	0.2742		0.382
Reputation for non cooperative peers	0.5960	0.5952	0.5919		0.605
Reputation for altruistic peers	0.6748	0.6777	0.7288		0.622
R	0.486	0.480	0.376	0.652	0.614

Results when we vary the fraction of malicious peers are given in Fig. 3.8. We can see that it is still possible to detect the attacker with all algorithms especially for CredTrust-trust and PerronTrust. For CredTrust and averageTrust , when the fraction of malicious peers reaches 44%, it becomes difficult to detect the attacker chosen by the coalition since its reputation is above the threshold of decision is 0.5.

Observe that CredTrust and AverageTrust give almost the same estimation of reputation for the chosen attacker.

Suppose that we want to compare our three algorithms to EigenTrust and AverageTrust in term of RTC (Ratio Trust Coalition) $= \frac{Trust\ of\ the\ chosen\ attacker\ peer}{Trust\ of\ altruistic\ peer}$.

Results are shown in Fig. 3.9 where we consider the three algorithms CredTrust-trust, PerronTrust and CredTrust compared to EigenTrust and AverageTrust, with different densities of malicious peers.

Observe that CredTrust and AverageTrust show high value of RTC because the attacker evaluates peers in a consistent way (like an altruistic peer) in order to increase his credibility. As CredTrust is only based on credibility the gap between altruistic and the attacker is small.

However, EigenTrust, PerronTrust and CredTrust-Trust are performing well but CredTrust-Trust considering credibility and trust gives better value of RTC.

3.4.3.7 Rational peers' behaviors: Non cooperative peers

Assuming that non cooperative peers give 0.5 to all peers in the system. Inverting peers evaluate the non cooperative peers with 0.7, the altruistic with 0.2 and others with 0.8. The altruistic peers evaluate non cooperative peers with 0.6, the inverting peers with 0.2 and other altruistic with 0.8.

We will suppose that 20% of the peers are inverting peers, 20% are non cooperative peers and 60% of the peers are altruistic. Also, we consider the ratio R as follows:

- $R = \frac{Trust\ for\ inverting\ peers}{Trust\ for\ altruistic\ peers}$

The obtained results are shown in the table 3.2.

In this scenario, we affirm that:

- the reputation assigned to inverting peers is clearly less important in CredTrust-trust than in PerronTrust, CredTrust and AverageTrust even with non cooperative peers in the system,
- the reputation assigned to altruistic peers is more important in CredTrust-trust than in PerronTrust, CredTrust and AverageTrust even with non cooperative peers in the system,

- the reputation assigned to non cooperative in the CredTrust-trust converges to trust given by altruistic. Being non cooperative does not mean being malicious or altruistic. Those peers adopt optimization strategy to save resources. Its reputation is not high and not low.
- R is less important in CredTrust-Trust than in EigenTrust, AverageTrust, CredTrust and PerronTrust.

3.4.3.8 Rational peers' behaviors: less efficient peers

In this attack, we have byzantine inverting peers and rational less efficient peers. This means that we simulate the inverting attack and we suppose that in the system we have efficient and 20 % less efficient rational peers. Less efficient peers are cooperative but they are not able to evaluate correctly malicious behaviors: they give 0.4 as an estimation for malicious behaviors (0.8 for altruistic and other less efficient peers). In addition, we have efficient altruistic peers that evaluate correctly other peers: estimating inverting peers with 0.2 and others with 0.8.

First, let us focus on the credibility seen in CredTrust and CredTrust-Trust. In Fig. 3.10, we show that the credibility of the altruistic efficient peers are more important in CredTrust-trust than in the CredTrust. We favorite these peers in the computation of the reputation scores in the system.

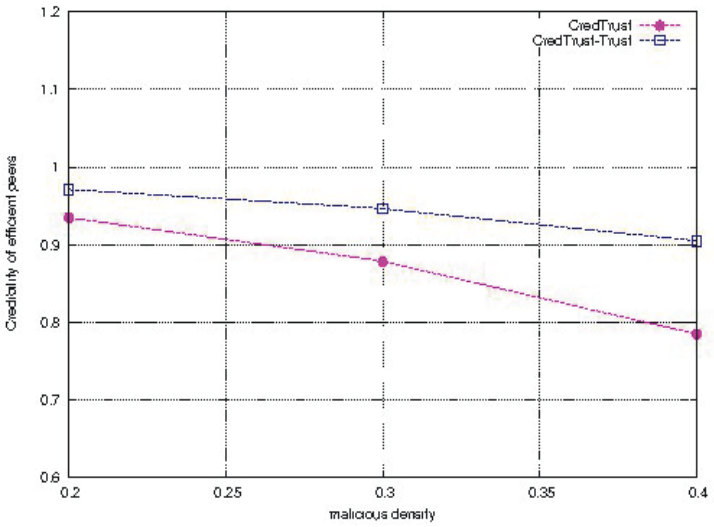

Figure 3.10: Comparison of credibilities

Second, results of the reputation of malicious peers under this efficiency problem are shown in Fig. 3.11. We notice that CredTrust-trust outperforms PerronTrust, CredTrust and AverageTrust. This model gives the most significant reputation to potential malicious peers under 40% of malicious ones. This is due to the credibility effect. In fact, the credibility of less efficient peers is considered in the computation of the reputation. Hence, the model considering credibility can efficiently distinguish efficient peers from less efficient ones.

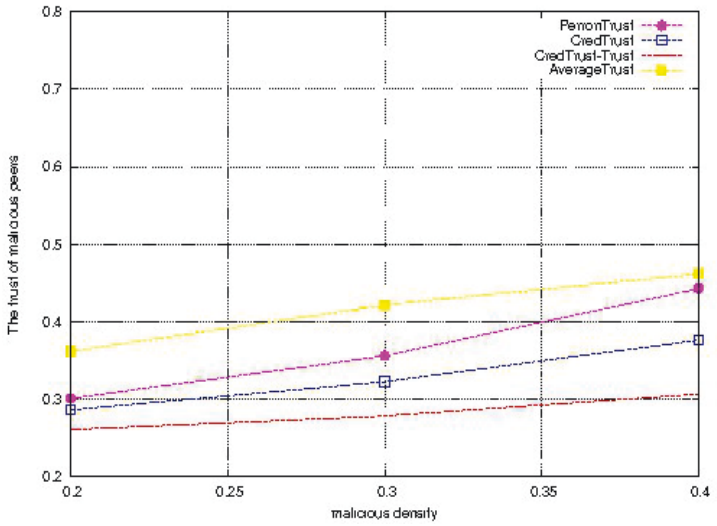

Figure 3.11: Comparison under efficiency problems

Third, suppose that we want to compare our three algorithms to EigenTrust and AverageTrust in term of $RTL(RatioTrustLesseffficient) = \frac{Trust\ of\ byzantine\ peers}{Trust\ of\ altruistic\ peer}$. Results are shown in Fig. 3.12.

We can notice that CredTrust-trust outperforms PerronTrust, CredTrust, EigenTrust and AverageTrust. In this case, we can notice again the benefit of credibility to make clear decisions between behaviors. So CredTrust and CredTrust-Trust outperform PerronTrust, EigenTrust and AverageTrust.

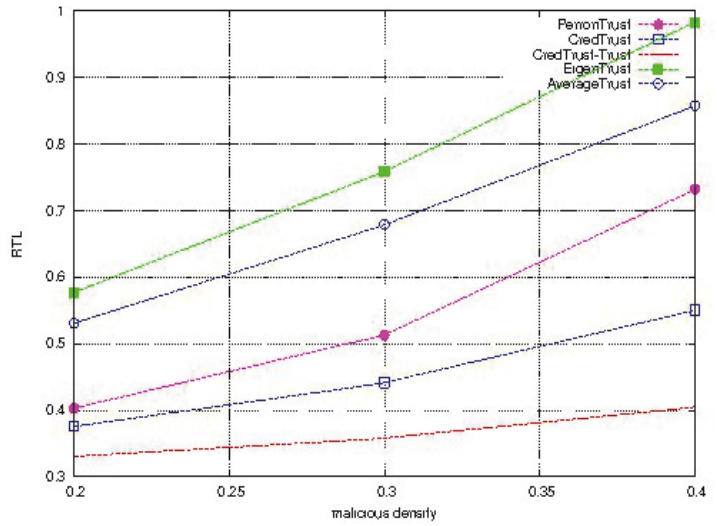

Figure 3.12: RTL under under efficiency problems

3.4.3.9 Pre-trusted peers benefit under inverting peers attack

We analyze the performance results of the two approaches described in the previous section. We show by simulation the benefit of pre-trusted peers in reputation calculation. Then, we compare the two new models with CredTrust-trust our previous preferred model under illustrated attacks. First, we consider especially CredTrust-Trust algorithm with pre-trusted peers and CredTrust-Trust algorithm with pre-trusted peers and we compare them under possible attack on trust and reputation systems.

We suppose here that all peers cooperate in the evaluation of reputation values. the reputation values given by peers in the system are as follow:

- inverting peers estimate inverting peers with 0.7 (or an other very high value) and altruistic peers with 0.1 (or an other very bad value),
- altruistic peers estimate byzantine peers with 0.1 and others with 0.9.

We consider that we have 49 peers in the privilege systems and 10 trusted peers with PTV=0.9.

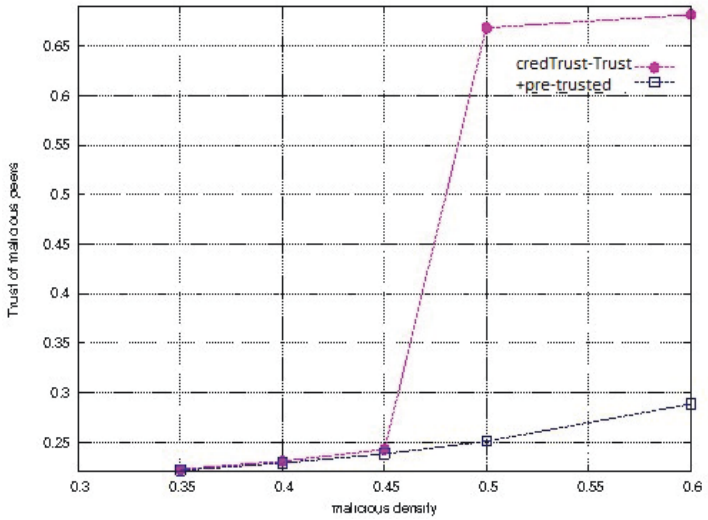

Figure 3.13: Trust under Byzantine peers' behaviors: inverting peers

The reputation values estimated for inverting peers are shown in Fig. 3.13.

When the proportion of malicious peers is about 10%, the three algorithms give almost the same estimation of malicious peers. when the fraction of malicious peers increases, the reputation values increase for the three models. Observe that CredTrust-trust show high value of malicious peers comparing to methods using the knowledge (pre-trusted). CredTrust-trust with pre-trusted peers are performing well.

3.5 Conclusion

In this chapter, we proposed new algorithms to detect rational and Byzantine (malicious) peers in the context of hybrid cloud. We considered a dynamic simple trust

as a peer to peer evaluation. Three models are compared under different cheating strategies. First, PerronTrust, a model considering the computation of reputation based on Perron algorithm. Second, CredTrust improves the first approach by introducing the concept of credibility when the reputation vector is updated. A third model, CredTrust-trust is proposed by combining the reputation and the credibility parameters in the iterative updating process.

Simulations under different attacks on the trust and reputation system and comparison with two well-known existing works are performed. The results of the first part of experiments show that the proposed model selects the dependable and reliable peers. Second part of simulations confirmed also that the credibility is well suited to clarify the behaviors of malicious (byzantine) and rational peers.

We also noticed that many collective attacks and compromised peers can be clearly detected and avoided with high and clear accuracy. This is particularly true when CredTrust-trust is used. It outperforms the well-known EigenTrust scheme and AverageTrust significantly. The combination of trust and credibility improves the performances of the method in a very significant way. Moreover, the complexity of the algorithm is very limited allowing it to be used in large scale systems.

Our framework results can help in making decision on whether to purchase execution in resources from an unknown supplier or not.

In the next chapter, we validate the efficiency of our models in two use cases, the hybrid computing and the peer to peer file sharing. We will propose some amelioration of our models based on some knowledge about peers.

Chapter 4

Refinement and uses cases : hybrid cloud and Peer to Peer file sharing

Contents

4.1 Introduction

To start, it is good to distinguish between:

▪ distributed architectures in their technical infrastructure (as the case of Peer to Peer or Cloud Computing) and,

▪ distributed architectures in their content - produced or developed in a distributed approach.

For the reasons of clarity and simplicity, we focus on distributed architectures infrastructure level.

We analyze the level of centralization of the architectures, which can be located on a continuum between total centralization with a hierarchical governance. In these architecture, resources are controlled and managed by a central entity (as is the case

of cloud computing) and totally decentralized with egalitarian governance in resource controlled, and managed by a balanced system of peers (as for most of P2P networks).

In the previous chapter, we proposed four methods to protect against attacks on the trust management system. Models are evaluated in an hierarchical way where an entity aggregates the reputation of the peers. Our defense mechanisms were analyzed but the question now is in which case trust will be useful. We need a realistic example to see how the architecture is applied and how it works. We are proposing in this chapter a trust based resource allocation method in hybrid cloud computing and trust-based file sharing in P2P network. This contribution is divided into two parts. In the first part, Our job allocation uses reputation of users while allocating the resources in hybrid cloud computing. This is important when resources are distributed in the network like in our case. In the second part, we use the Query Cycle Simulator, a file sharing P2P networks simulator. We modified the simulator to support our model. CredTrust-Trust algorithm is used to assign peer trust values based on the success of a file upload. So, the purpose is to show how useful and efficient it can be.

4.2 Validation in hybrid cloud computing

4.2.1 A refined trust management model

Hybrid cloud computing is very useful for improving the distributed applications performance. However, it is difficult to manage risks related to trust when collaborating with unknown and potentially malicious peers. Trust management for this kind of transactions needs special processing. This processing should be based on the reputation because it will make it efficient and significant.

The main focus of this part is to develop a trust model, which will help in dynamically assessing the reputation of the intrusion detection systems in dynamic network. The important characteristic of our proposal is that we have identified two major factors (correctness of the execution and time taken to do the job), which will influence the performance of peers. In this trust management scheme, a peer's trustworthiness is calculated by the feedback it gets for each of its transactions with the other peer. If the peer performs consistently well, its trust increases gradually.

The trust of peers can be estimated based on local observations of their behaviors. So, after the interaction between peers, each peer will give a trust value corresponding to the verified peers.

This trust value in the context of execution on cloud (explained in the previous chapter) is based on each interaction in the past. We can notice the importance of some factors in the evaluation process: the performance of the peer in terms of time taken, the correctness of the returned results and the crashes experience. Cloud peers can be prone to errors. When each peer ensures that the returned results are correct and the time is respected, the risks are reduced. If the results are not correct, the task is resubmitted to other peers. For that, we construct a model to manage the trust.

Malicious peer may not respect this model and give wrong values. Therefore, we use the pair trust score $T_{ij}(t)$ which is an evaluation of machine $\boldsymbol{i}$ to machine $\boldsymbol{j}$ at

time $\boldsymbol{t}$ explained in Eq.4.1. It is represented as a real number in the range of [0, 1] where 1 indicates a complete good reputation, 0.5 ignorance, and 0 bad reputation. The reputation value is the weighted average of values related to correctness T^c (Eq 4.2) and efficiency in term of time T^t(Eq 4.3).

$$\mathrm{T}_{ij}(t) = w_c.T^c_{ij}(t) + w_e.T^e_{ij}(t) \quad (4.1)$$

$w_c + w_e = 1$

w_c, w_e are the weights associated with each component.

$$\mathrm{T}^c_{ij}(t) = \begin{cases} p_c.T^c_{ij}(t-\delta\mathrm{t})+(1\text{-}\mathrm{p}_c).T^{deg}_{ij}(t), & \text{if i checks j;} \\ T^c_{ij}(t-\delta\mathrm{t}), & \text{else.} \end{cases} \quad (4.2)$$

$T^{deg}_{ij}(t)$: refers to the degree of correctness of the execution of jobs seen by i when verifying j at time t.

$$\mathrm{T}^e_{ij}(t) = \begin{cases} p_e.T^t_{ij}(t-\delta\mathrm{t})+(1\text{-}\mathrm{p}_e).T^{eff}_{ij}(t), & \text{if i checks j;} \\ T^t_{ij}(t-\delta\mathrm{t}), & \text{else.} \end{cases} \quad (4.3)$$

$T^{eff}_{ij}(t)$: refers to the efficiency in term of time taken for the job evaluated by i.

The weights: p_c, and p_e are associated with each component for the correctness and the efficiency.

The result of the trust evaluation process is a matrix containing pair trust values between peers.

4.2.2 Trust-based reputation management with privilege parameter

In the previous chapter, the last model of CredTrust-trust, is proposed by combining the trust and the credibility parameters. This model shows a good performance and develops good defense mechanisms against attacks on reputation. In this part, we ameliorate and adapt this method to meet our requirements in term of job allocation performance. For that, we add a new concept which is the privileged peers.

Let us introduce **B** as the privilege matrix, where the portal gives different weights to peers in the system. To estimate whether a peer is really a good trust evaluator or not, it may be important to focus on what we mentioned about some special nodes. This is expressed through this privilege matrix by putting a value in the diagonal

decided by the portal based on the use case. So, we put in B the trust value of some complectly trusted node.

In the Alg. 7, we illustrate the use of the privilege matrix in the CredTrust-Trust described in the previous contribution, byzantine resistant trust-based reputation management.

Algorithm 7: CredTrust-Trust algorithm with privileged peers

Require: ε, N
Ensure: π
1: **while** $||Cred(t) - Cred(t-1)||_1 > \varepsilon$ **do**
2: **for** $i \in \{1, ..., N\}$ **do**

$$Cred_i = 1 - \left(\frac{\sum_{j=1}^{N} B_i |\pi_j - T_{ij}|^2}{\sum_{j=1}^{N} |B_i \pi_j - [1 - \pi_j]|^2} \right)^{\alpha}$$

3: **end for**
4: $\pi \leftarrow \frac{T^t . \pi}{||\pi||_1}$
5: **end while**

4.2.3 Performance of privilege based model in term of jobs loss

In the table 4.1, the parameters used in the following analysis are explained.

Table 4.1: Some notations

N	Number of machines including in the application
I	The vector containing the initial knowledge about peers' trust
JobsL1	Jobs loss without considering trust management
JobsL2	Jobs loss with considering credTrust-Trust
JobsL3	Jobs loss with considering credTrust-Trust and privilege
Capacity $_i$	Maximum of executed jobs in machine i
CapacityCred $_i$	Maximum of executed jobs in machine i after considering CredTrust-Trust
CapacityPriv $_i$	Maximum of executed jobs in machine i after considering CredTrust-Trust and privilege

Initially, regarding the initial trust about peers, we can estimate the initial jobs loss $JobL1$. It is the first estimation of the probability to execute jobs in malicious machines. It is estimated as the following :

$$JobL1 = N * \frac{\sum_i capacity_i}{\sum_i capacity_i} * (1 - I_i)$$

We have proposed previously models for the trust-based reputation management. In this chapter, we are focusing on the credTrust-Trust as it shows a good performance.

We compare it with the last proposed model : CredTrust-Trust with privileged machines. The comparison is based on the jobs loss. Let us focus on the last model as we have a new parameter "privilege". In our use case, it is suitable to have the privilege as the capacity of the machine to execute jobs. As we mentioned before, the privilege denotes the importance of the machines in the system. The capacity parameter is important to submit jobs in our system. Owing to this, machines having high capacity of execution are defined as critical ones. We want to focus on what is mentioned about those critical machines in the system. After the trust and reputation management (with models CredTrust-trust and credTrust-Trust with privilege), we obtain an order of machines based on the final vector of the reputation. To arrange the new capacity vector, based on the obtained order we begin with the capacity of the most trusted peer. When we use CredTrust-trust, we obtain the vector $capacityCred$. When we use the credTrust-Trust with privilege, we obtain the vector $capacityPriv$. An example to explain the construction of $capacityPriv$ is giving below in the table 4.2.

Table 4.2: Capacity vector

Capacity	10	5	7	14
Reputation (+priv)	0.4	0.6	0.1	0.35
CapacityPriv	5	10	14	7

Considering this information about capacity and the reputation, we start with the most trusted peer and we affect the maximum of jobs and we move to next and the next until the exhaustion of the jobs or the last peer in the system.

To estimate the jobs loss after considering the CredTrust-Trust, $JobL2$, we use the following formula :

$JobL2 = \sum_i capacityCred_i * (1 - I_i)$

To estimate the jobs loss after considering the CredTrust-Trust with privilege, $JobL3$, we use the following formula :

$JobL3 = \sum_i capacityPriv_i * (1 - I_i)$

As we want to show the benefit of reputation and the privilege in term of jobs loss, we compare the results of $JobL1$, $JobL2$ and $JobL3$. In our simulation, we vary the number of the malicious peers to obtain the Fig. 4.1 with different malicious densities. In the Fig. 4.1, we show that:

- JobL1 is more important than JobL2 and JobL3,
- the difference between JobL1 and JobL2 and between JobL1 and JobL3 is increasing when the malicious density is more important.
- JobL2 is more important than JobL3

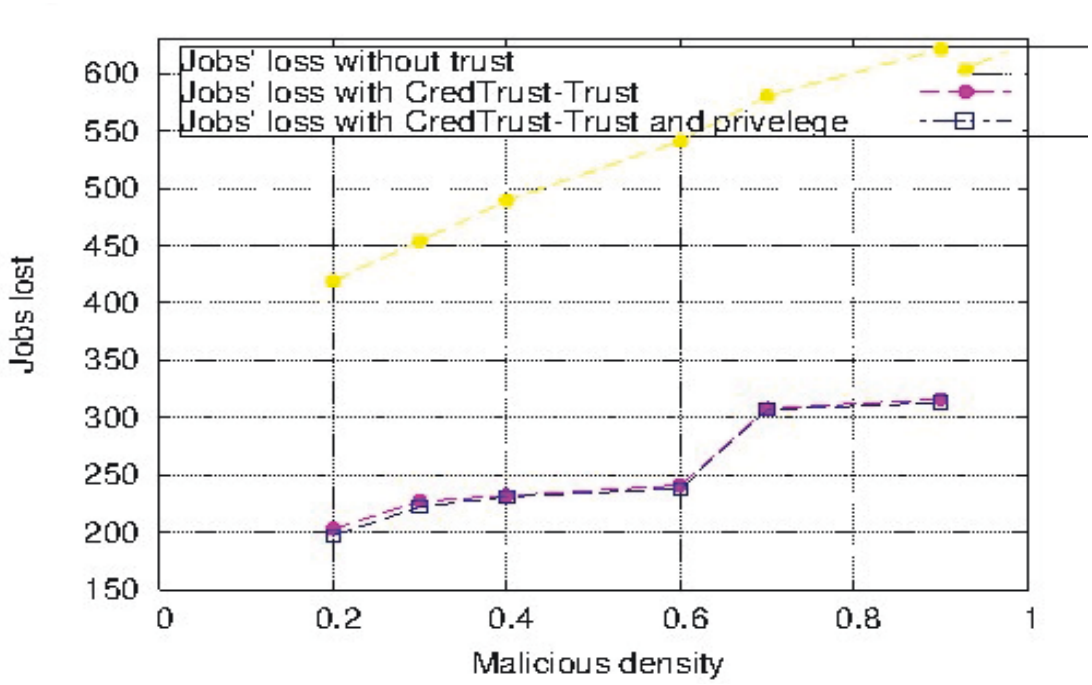

Figure 4.1: Jobs loss

We can conclude from the simulations and analysis that considering the reputation management models have a better impact on the system. In term of jobs loss, we observe the decrease of this number to almost the half compared to the loss without trust.

4.2.3.1 Performance of our reputation-based jobs affectation

The credibility idea explained in the previous chapter can be evaluated also in the case of jobs affectation. The experimental setup consists of ten resources (peers) controlled by the portal. A sample of 100 jobs were submitted to a portal with an assumption of only ten public clouds resources R1, R2,..., R10 are available. Suppose that R1 and R2 are malicious (possibility of malicious code insertion or else).

The results are shown in Fig 4.2. From this figure, it is obvious that the number of jobs allocated to public cloud resources without trust policy is 9 and 8 for malicious resources and between 7 and 15 for trusted peers. The selection of peers in the first phase is based on the free resources and without considering the trust value. In our proposal, the selection of resource is based on the trust value of the peers. We suppose to compare one trust and reputation management without credibility (PerronTrust) and one with (CredTrust-Trust). Hence for both, the requested job is assigned to the most trustworthy peers. For CredTrust-Trust, the number of jobs as submitted to malicious peers (R1 and R2) is (5 and 2) lower than for PerronTrust (7 and 3) and without trust (8 and 9). For CredTrust-Trust and PerronTrust, the number of jobs as submitted to trusted peers (R3 ... R10) is high.

This states that low trusted peers are provided with a less number of jobs than

reliable ones when we consider the trust value (for both PerronTrust and CredTrust-Trust).

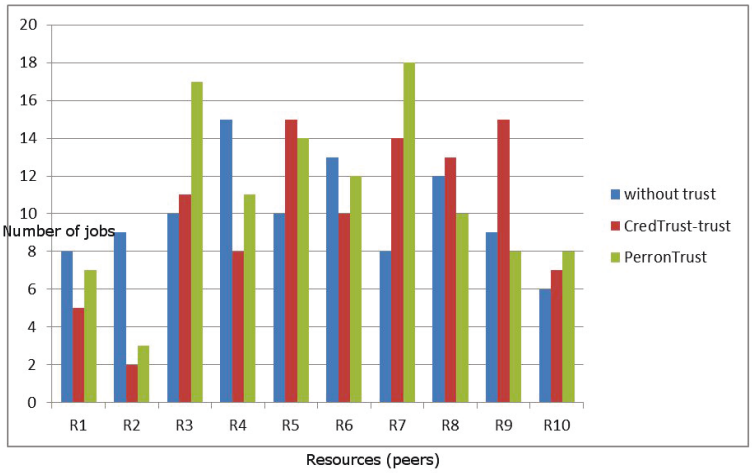

Figure 4.2: Benefit of trust in jobs affectation

4.3 Validation in peer to peer systems

We implemented our trust-based reputation management model using JAVA in peer to peer. We are planning to implement our centralized algorithm in the real file sharing systems. For that, we use the Query Cycle Simulator (QCS), which is used for file-sharing P2P networks networks. We integrated our model in the simulator. Let us first explain the P2P file simulator QCS, then present the integration of the trust option in the file sharing.

4.3.1 P2P file simulator presentation

We consider a typical interconnected P2P network. In this network, file-sharing peers are able to issue queries for files, peers can respond to queries, and files can be transferred between two peers to conclude a search process.

In the first interface, there are two groups of controls that allow you to setup a desired network configuration. The first group is the attribute panels, which contains command buttons. These buttons will display panels at the button of the view. The second group is the peer networks

In Fig.4.3, we can observe the second group of buttons to set up the network on the left. So, we can distinguish between all types of peers: pre-trusted, malicious and honest peers to configure the network. In the same figure, we present the topology and the links between neighbors.

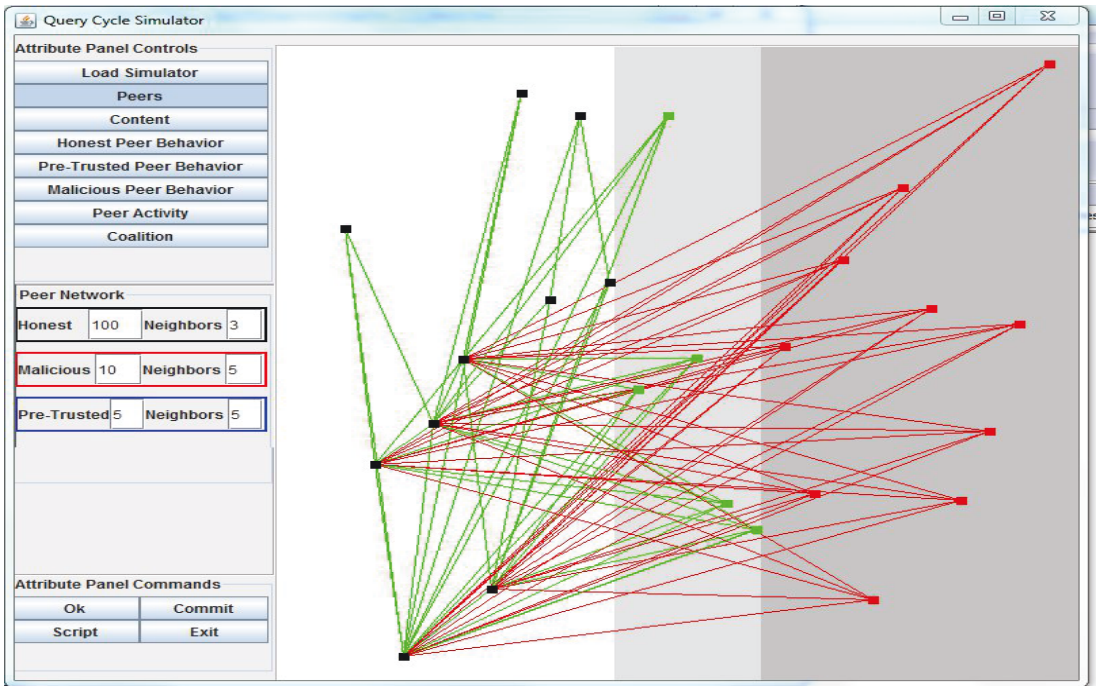

Figure 4.3: Principal interface

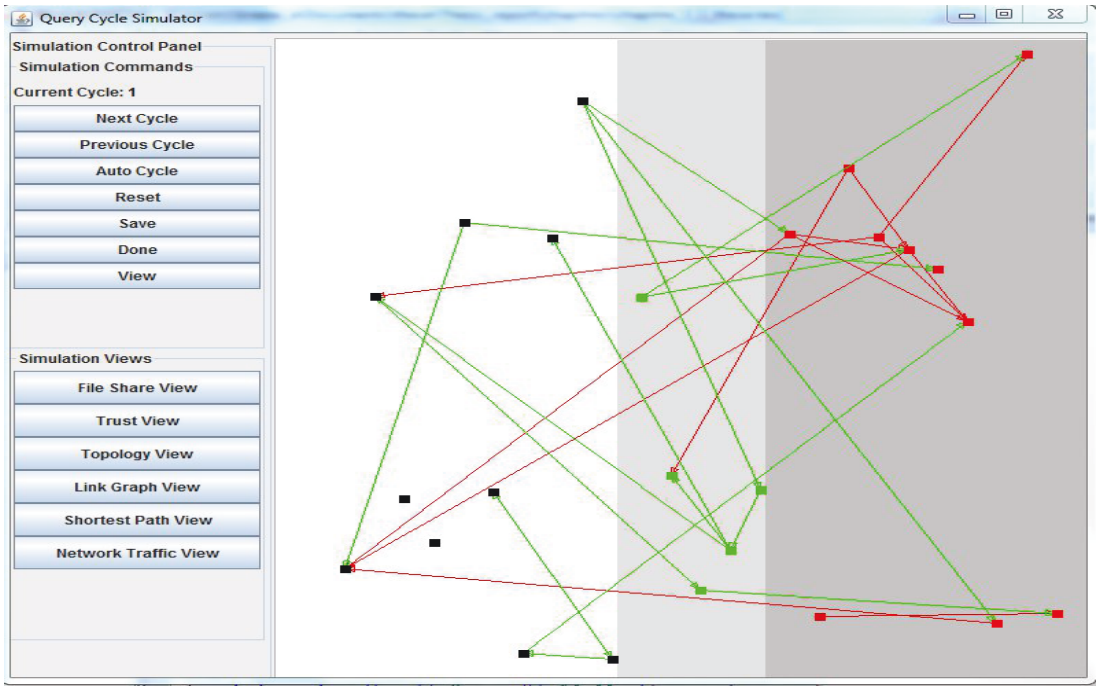

Figure 4.4: Simulation management

In Fig. 4.4, peers are differentiated using colors red, black and green:

- red for malicious peers,

- black for the honest peers
- green for the pre-trusted peers.

Links are differentiated using red and black colors:

- red lines for links including malicious peers,
- black lines for links including the honest peers and pre-trusted peers.

Our network consists of honest peers (normal nodes, participating in the network to download and upload files) and malicious nodes (adversarial nodes, participating in the network to undermine its performance). As described in the previous chapter, some honest peers in the network are appointed as pre-trusted peers.

The simulation of a network proceeds in simulation cycles: Each simulation cycle is subdivided into a number of query cycles. In each query cycle, a peer i in the network may be actively issuing a query, inactive, or even down and not responding to queries passing by. Upon issuing a query, a peer waits for incoming responses, selects a download source among those nodes that responded and starts downloading the file. The latter two steps are repeated until a peer has properly received a good copy of the file that it has been looking for. Statistics are collected at each node, in particular, we are interested in the number of authentic and inauthentic up- and downloads of each node. Each experiment is run several times and the results of all runs are averaged. Each experiment is run until we see the convergence to a steady state.

4.3.2 Benefit of our proposed reputation management in P2P file sharing

In this part, we suppose that the sharing files between peers is decided based on the reputation. In Fig 4.5, we show that in the cycle 4 of the file sharing, the reputation of malicious peers is very small (represented on red). But, the reputation of pre-trusted and honest peers is increasing. Those peers are represented in green for the pre-trusted peers and in black for the honest peers.

In the simulation, to prove the benefit of reputation, malicious peers are always providing an inauthentic file when selected as a download source. We simulate a network consisting of 53 good peers, 5 of them are pre-trusted peers.

The experiments are on a system where download sources are selected based on our trust values shown previously and where download sources are chosen randomly from the set of peers responding to a query. In each experiment, we add a number of malicious peers to the network such that malicious nodes contribute between 10% and 40% of all peers in the network. The results are illustrated in Fig. 4.6.

As y axis represents the inauthentic files, bars depicted this fraction of inauthentic files downloaded in one simulation cycle versus the total number of files downloaded in the same period of time. The results are averaged over the last 10 query cycles in each experiment. We can observe that the fraction of inauthentic files is increasing

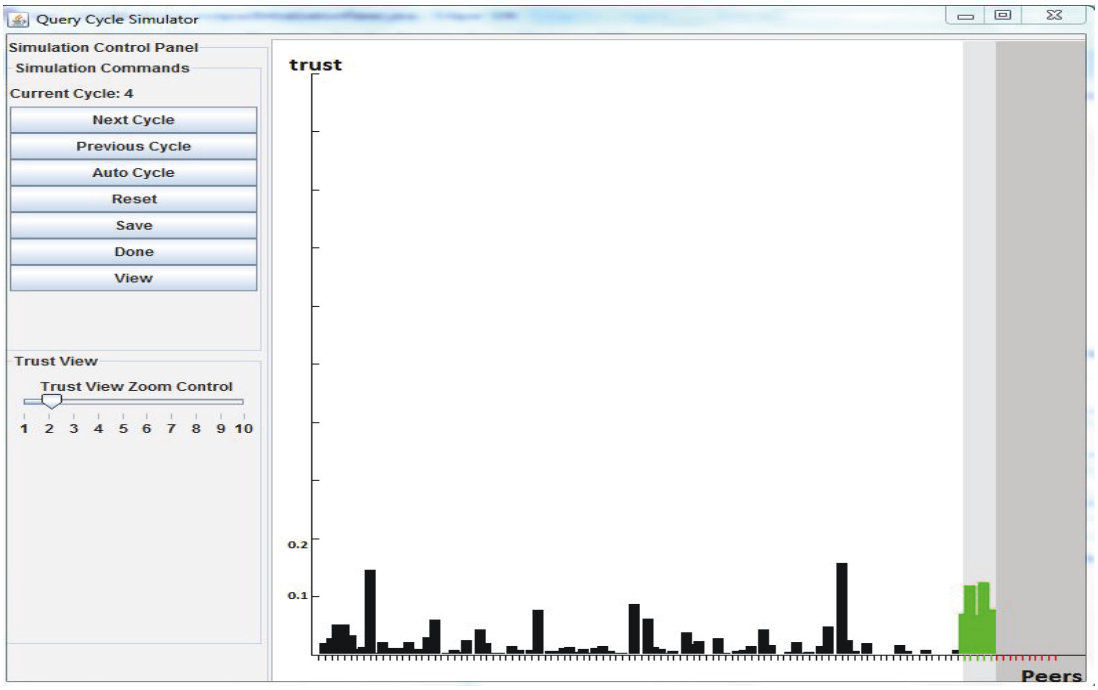

Figure 4.5: Trust of peers

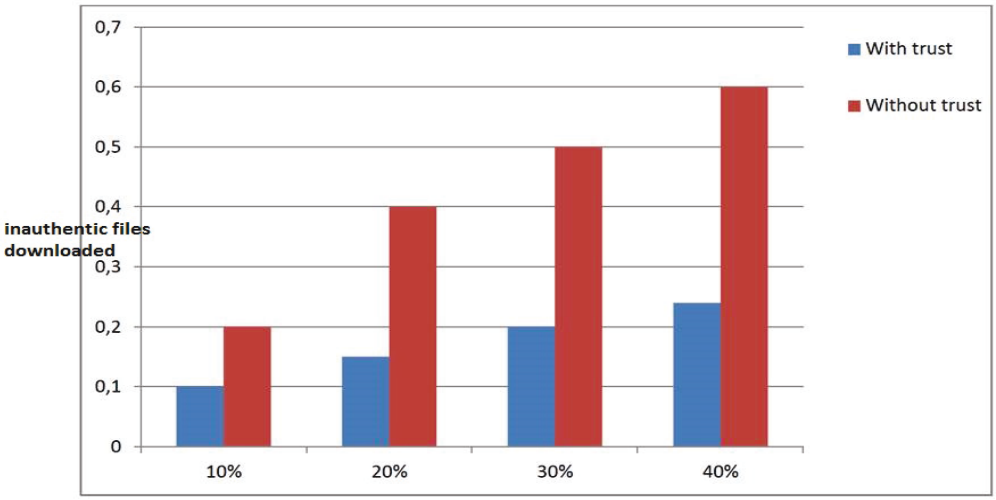

Figure 4.6: Reduction of inauthentic downloads

rapidly if the selections are not based on trust and reputation. But, for the second case the fraction can reach only almost 25% for 40% of malicious peers.

Without trust and reputation management, malicious peers succeed in inflicting many inauthentic downloads on the network. Yet, if our scheme is activated, and due to low reputation values of malicious peers, they are rarely chosen as download source, which minimizes the number of inauthentic file downloads in the system.

In most distributed networks a similar approaches help improving reputation. We analyze in term of scalability of our system to ten of thousand of peers in some cases. In this case, we have a very large matrices and sparse matrix.

4.4 Conclusion

In this contribution, we considered a composite trust estimation model derived from correctness and efficiency in terms of time. To make our ideas more clear and refined, we tried to validate the importance of our new refined idea in term of jobs loss. Then, we validate the credibility idea in the context of cloud resource allocation. We simulated the file sharing problems using our model. We show that credibility idea has benefit in term of minimizing the inauthentic uploads in the file sharing system.

Chapter 5

Game theory approach to optimize trust-based cooperative intrusion detection

Contents

5.1 Introduction

In the previous contribution, we supposed that peers participate in the evaluation of the reputation. Using game theory, we want to optimize the participation in a distributed system where actors run some kind of trust evaluation in between. So, in this chapter, we try to evaluate our proposed optimization in the context of distributed intrusion detection.

5.1.1 Trust-based cooperative intrusion detection

In this part, we briefly enumerate some contributions on intrusion detection and trust management. Host-based Intrusion Detection System (HIDS) [85] consists of an agent that identifies intrusions by analyzing activities on the host and its current state. An

adapted version of this system will be used in our detection algorithm. In [86], the intrusion detection in the cloud computing is considered. In [87], a detection system is proposed to reduce the impact of denial-of-service (DoS) attacks. HIDS have been widely used to detect malicious behaviors of nodes in heterogenous networks.

Research in intrusion detection studies the effectiveness of intrusion detection and how to handle attacks against the intrusion detection system itself. We investigate the cooperative approaches between different detection units. Li et al. [88] have used mobile agents for developing a coordinated distributed intrusion detection scheme for ad hoc networks. In the proposed scheme, the mobile nodes are divided into different clusters. The cluster-head acts as the manager of nodes. Each cluster-member node runs a host monitor agent to detect network and host intrusions using intrusion analyzer and interpretation base. The assistant agent running on a cluster-head is responsible for collecting intrusion-related data from the cluster-member nodes.

Trust and security are challenging problems in cooperative intrusion detection. In the literature, few contributions combine trust and intrusion detection together. Wang et al. [98] proposed an intrusion detection mechanism based on trust (IDMTM). This work considered both evidence chain (main malicious behavior forms) and trust fluctuation. They provided more accurate decision with low false alarms. Ebinger et al. [99] introduced a cooperative intrusion detection method for Manets based on trust evaluation and reputation exchange. The reputation and confidence are combined with trustworthiness to improve intrusion detection. The contribution didn't however consider dynamic Manets environments.

Collaborative intrusion detection can be more secure with a framework using reputation aggregation as an incentive. The problem of incentives and efficiency are well known problems that can be addressed in such collaborative environment. In this chapter, we propose using game theory to improve detection and optimize intrusion detection systems used in collaboration. The main contribution of this chapter is that the reputation of HIDS is evaluated before modeling the game between the HIDS and attackers. Our proposal has three phases: the first phase builds reputation evaluation between HIDS and estimates the reputation for each one. In the second phase, a proposed algorithm elects a leader using reputation value to make decisions. In the last phase, using game theory the leader decides to activate or not the HIDS for optimization reasons. HIDS systems degrade when maximum security is applied at all times. Hence, to detect attacks and minimize the consumption of resources due to monitoring, an iterative, efficient and dynamic security system is needed.

5.1.2 Game theory

Game theory is a set of modeling tools that provide a mathematical basis for the understanding and the analysis of interactive decision-making problems for actors involved in situations with conflicting interests. It has been applied in real games, economics, politics, commerce and recently in telecommunications and networking. Game theoretic platform is suitable for modeling security issues such as intrusion detection and prevention detection. A game consists of three components:

- a set of rational players that interact to make decisions,

- a set of possible actions (strategies) for each player,
- a set of utilities that are functions of action profiles that determine the outcome of the game. In other words, the utility function assigns a value to each possible outcome; higher utilities represent more preferable outcomes.

a_i is the action of the player i. Each player aims to maximize her/his utility function. For political scientists for example, actions may be electoral platforms choices and votes.

A game model is generally appropriate only in scenarios where decisions of each actor impact the outcomes of other actors. In a system involving several players, we can distinguish between two types of games where players may be cooperative or competitive. In a cooperative game, the problem may be reduced to an optimization problem for which a single player drives the system to a social equilibrium. A criterion used in game theory to express efficiency of such equilibrium is Pareto efficiency [115]. A strategy profile is called Pareto efficient if no other strategy exists such that:

- all users do at least as well
- at least one user does strictly better.

  In a non-cooperative game, each player selfishly chooses his (her) strategy. In this case, if an equilibrium is reached, it is called a Nash equilibrium. It is the most well-known equilibrium concept in game theory and is defined as the point from which no player finds it beneficial to unilaterally deviate.

In a wireless system, the players may be mobile nodes, networks or services. Actions may include the choice of a modulation scheme, a flow control parameter, a power level, a bandwidth amount or any other factor that is controlled by the network, the node or the service. These actions may be constrained by technical capabilities or resource limitations or by rules or algorithms of a protocol. However, each player in this context will dispose of some leeway to set the appropriate parameters to his (her) current situation or even to totally change his (her) mode of operation. These players are then autonomous agents that are able of making decisions about bandwidth allocation, transmit power, packet forwarding, backoff time, and so on. As stated before, players may cooperate or not. In the context of wireless networks, nodes may look for the "greatest good" of the network as a whole, they may also behave selfishly, seeking their own interests or they may even behave maliciously, aiming to damage the network performance for other users.

The rest of the chapter is organized as follows. Section II exposes some related works on game theory for intrusion detection and a comparative study. In Section III, our model and different possibilities are described . Section IV shows some simulations results and analyzes. Finally, the chapter concludes and lists some possible future extensions for future work.

5.1.3 Intrusion detection and game theory

Intrusion detection research mainly rely on progressive knowledge of protocol flaws and system weaknesses. There is an interest in collaborative frameworks and trust

Table 5.1: Comparative study

Game solutions	Contributions(+)	Limitations (-)
Dynamic non cooperative game [103]	Maximize the effective payoff by minimizing the cost due to false alarms and missed attacks	No consideration about selfish node activity No consideration about malicious node coalitions
Bayesian game [107]	To detect the intrusion based on the belief updated by the regular node about its neighbor	Does not consider the scenario when attackers come in a group
Non cooperative game [103]	Increase its monitoring probability by reducing the utility of the malicious coalition	Resource consumption
Stackelberg game [108]	Defender's strategy can be viewed as a function of attacker's strategy.	Complicate equilibrium
Bayesian Stackelberg game	The leader is uncertain about the types of adversary it may face	Heuristic methods to solve

management to federate intrusion detection systems [101]. In the literature, few contributions combine trust and intrusion detection together. Wang et al. [98] proposed an intrusion detection mechanism based on trust (IDMTM). This work considered both evidence chain (main malicious behavior forms) and trust fluctuation. They provided more accurate decision with low false alarms. Ebinger et al. [99] introduced a cooperative intrusion detection method for MANETs based on trust evaluation and reputation exchange. The reputation and confidence are combined with trustworthiness to improve intrusion detection. The contribution didn't however consider dynamic MANET environments.

The early model of game-based intrusion detection goes back to 2002 [96]. It was a general-sum stochastic game between an attacker and the administrator. In [111], a multi-stage intrusion detection game is implemented with dynamic update of the belief. In [97], authors want to incorporate a practical scheme with decision and control developed with game-theoretic tools. Previous works also proposed to establish belief between nodes using bayesian games [108]. But this technique can create overhead in the network. In [110], we proposed to use reputation to update trust between nodes on hybrid cloud computing. It is a lightweight mechanism using that gives dynamic evaluation. Obtained trust scores will help the execution of application in cloud and to guide them.

Several related contributions about game theory in intrusion detection are illustrated and summarized in Table 5.1.

Table 5.2: Lead or Follow

Strategies	Advantages
The attacker as leader and defender as follower	The attacker can control the strategy of the defender. The defender will take reasonable reactions according to the attack strategy of network attacker The attacker will adjust its strategy in reaction to the defender's strategy.
The defender as leader and attacker as follower	The defender can control the attacker side. Performance parameters can influence the leader.

Recent interest focused on the leader-follower Stackelberg game. In this game, the interaction between attackers and defenders is a non-cooperative Stackelberg game, in which both the leader and the follower choose their strategy such that both sides try to maximize their payoff. This game has been successfully implemented in several real-world domains, including Los Angeles International Airport (ARMOR program) and IRIS program in use by the US Federal Air Marshals. However, they failed to address how a leader should compute his strategy given uncertainty about the follower' capability. Two cases can be illustrated: "attacker as leader and defender as follower" or "the defender as leader and attacker as follower". We briefly compare the "lead or follow" advantages in Table 5.2. We can conclude that it is not trivial to choose who is the leader and who is the follower.

5.2 A three phase proposal

In this section, an introduction to our system is proposed. We start with the network, then we describe the three steps of our proposal.

We consider a network G(v) where $v = S_{HIDS}, S_A$:

- S_{HIDS}: is the set of defender nodes equipped with an host-based intrusion detection system.
- S_A: is the set of attackers. In this chapter, they can be malicious nodes.

Our proposal has three steps: the first step builds global reputation relationship between HIDS. The reputation estimated for each node can guide the game-based model as we use the reputation as a gain. In the second step, based on this evaluation of reputation and the estimated resources, the algorithm elects a leader which will play the game, calculate the global reputations and decide for activating HIDS. In the third step, a game theory model helps HIDS in the cooperative detection model.

In Fig.5, we show the evolution of the game-based framework. After the peer-to-peer evaluation, the algorithm outputs local reputation evaluations that feed the dynamic trust system. With this first hand evaluation, we can detect abnormal behaviors or internal attacks between HIDS in the system. Also based on this first evaluation of trust and resource management, we elect an efficient leader. This leader

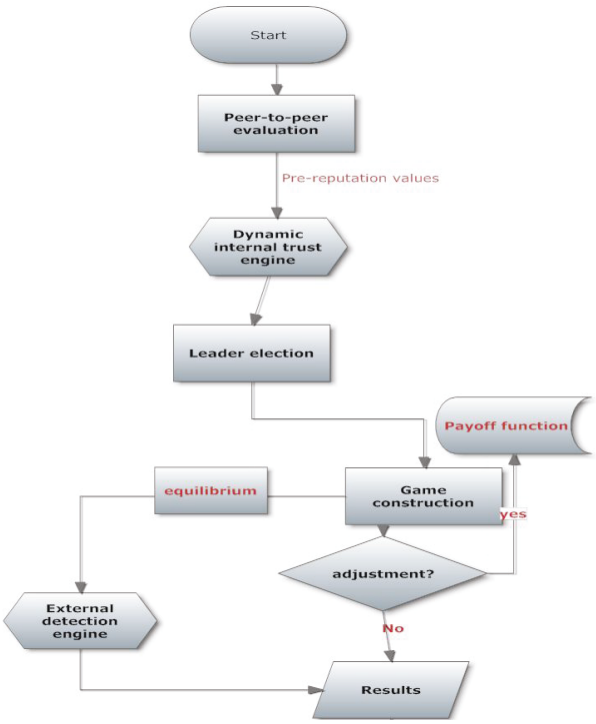

Figure 5.1: Three steps flow diagram

will make monitoring decisions in the game. We construct payoff functions to determine the optimal strategies of the attacker and the HIDS. Based on equilibrium results, we can launch the HIDS for detecting attacks.

5.2.1 The first step: Building reputation between HIDS

Without incentives, a node can adopt cheating behavior by lying about its remaining resources and its reputation. Our proposal, in the sense of collaborative intrusion detection, gives incentive to nodes as reputation values. Due to the existence of various kinds of malicious peers, it is obvious that reputation evaluation between peers must be set up. The local reputation value is measured by evaluation done between HIDS. Global reputation estimation is calculated by the referrals from all peers.

Reputation model Let us first introduce some definitions and notations needed for our models. Then, we describe the proposed algorithms to compute the global reputation.

Definition 11 *Let T be a local reputation matrix. It is initialized to be 0.5.*

Definition 12 *$v_{ij}(t)$ is the direct evaluation of behaviors.*

$v_{ij}(t)$ is updated through verifications like challenge sent periodically: it represents the fraction of positive results when peer i verifies peer j. For example after verifications between two peers 1 and 2 with 4 positive results and 5 negative results, the evaluation score is 4/4+5=0.44. So, $d_{ij}(t) = 0.44$.

Definition 13 *δt is the update period*

$||.||_1$ and $||.||_2$ respectively stand for the l_1-norm and l_2-norm of vectors.
T^t is the transpose of the matrix T.

Definition 14 *Let π be the global reputation vector of size N. Initially $\pi_i = 0.5$ where $i = 1, 2, ..., N$. It is represented as a real number in the range of [0, 1]*

Definition 15 *ε is the convergence threshold.*

Our proposed algorithm is a power method. Notice that the power method is used in different areas including, for example, the computation of the PageRank of web documents [112]. PageRank ranks a page according to how many other pages are pointing to it. Google's search engine is based on this PageRank algorithm. [113] is the most similar to the reputation mechanism in this work. This approach computes a global trust metric using system-wide information. However, authors make constraint that $T_{i1} + T_{i2} + ... + T_{in} = 1$. In [113] the notion of transitive trust that addressed the collusion problem is solved by assuming that there are pre-trusted peers in a distributed network. However, our model computes the global reputation in an elected peer considered as trusted leader. In addition, the local reputation $T_{ij}(t)$ value is evaluated differently following Eq.5.1.

$$\mathrm{T}_{ij}(t) = \begin{cases} p.T_{ij}(t-\delta t) + (1-p).v_{ij}(t), & \text{if i verifies j;} \\ T_{ij}(t-\delta t), & \text{else.} \end{cases} \tag{5.1}$$

In eq. (5.1), we assign more weight to recent interactions and less weight to the previous ones. So, p is used for that purpose.

Using the reputation matrix T, and the current global reputation vector, we get a new approximation of the trust of each peer j through a combination of the reputations of j as Eq.5.2:

$$\pi_j \leftarrow \frac{\sum_{i=1}^{N}(T_{ij}\pi_i)}{||\pi||_1} \tag{5.2}$$

If a peer i has a high global reputation score, then it is normal to give more importance to the reputation values that it is assigning to other peers. Writing Eq.5.2 for each peer leads to Eq.5.3:

$$\pi \leftarrow \frac{T^t \cdot \pi}{||\pi||_1} \tag{5.3}$$

The global reputation will then be iteratively computed by repeating Eq.5.3 until the reputation vector becomes stable. The convergence of Alg.8 is based on the Perron-Frobenius theorem that we recall here for sake of completeness. The Perron Frobenius theorem asserts that a real square matrix with strictly positive entries has a unique largest real eigenvalue and that the corresponding eigenvector has strictly positive components.

Algorithm 8: PerronTrust algorithm

Require: ε, N
Ensure: π
1: Retrieve local reputation values (in T)
2: Initialize the reputation vector π
3: **while** $||\pi(t) - \pi(t-1)||_1 > \varepsilon$ **do**
4: Calculate the global reputation vector : $\pi \leftarrow \frac{T^t \cdot \pi}{||\pi||_1}$
5: **end while**

We have demonstrate the convergence of this algorithm in the chapter three (Byzantine resistant trust-based reputation management)

5.2.2 Second step: Election

In this part, we propose to elect a honest node with normal behavior and good level of resource. This feature is considered in prior phase through performing collaborative evaluation between neighboring nodes. The leader should have high trust value view. In the election process it may occur that one node has maximum resource to monitor but it hasn't high trust value or vice versa. In this situation, we will choose nodes with high trust score because the monitoring algorithm in our method is cooperative. We propose the Alg. 9 to select the leader. In this algorithm, let N_j be the set of all neighbors of node j (if the euclidean distance between two nodes is less than the transmission range, they are considered as neighbors). Some notation and definitions are given below:

rep_0: the threshold of reputation value,
R_{IDS}: the resource needed to have an HIDS,
A_k: the ratio of resource of node k and the time, R_t be the total resource,
R_{exe}: the resource needed for executing another task,
T_t: the total time, T_{exe} be the time needed for executing another task
ID_k : the identity of the node k.

In the Alg9, nodes start the election by sending a specific message. Then, each node calculate the ration of the resources and the time. When a node receives the start message, it sends a Hello message. The election is based on the reputation and a voting mechanism illustrated by specific procedures in the algorithm.

Algorithm 9: Leader Election

All send Start-election
for $k \in 1..M$ **do**
  if "$((R_t^k - R_{exe}^k) > R_{IDS})$" **then**
    $A_k = \frac{R_t^k - R_{exe}^k}{T_t^k - T_{exe}^k}$
  else
    $A_k = 0$
  end if
  if "Node j received Start-election from N_j" **then**
    Send hello(ID_k, A_k)
  end if
  for $i = 1$ to $i <= |N_k|$ **do**
    if "$(\pi_i > rep_0)$" **then**
      if "$\pi_i > rep_0 \;\; A_i > A_k$" **then**
        send vote (k,i)
        leader(k)=i
      else
        Mark (node i)
      end if
    end if
  end for
  Node i sends ACK to all nodes (all vote received)
end for

5.2.3 Third step: Detection based on reputation and game theory

In this step, we model and optimize our problem for intrusion detection of external risks using game theory. It is a two-player, non-zero-sum, non-cooperative, iterated game, in which the number of rounds depends on the use case explained in [109]. An attacker node can choose to attack or not to attack, whereas a defending node can choose to monitor (acts as an HIDS) or not to monitor (private information). The objective of the attacker is to attack without being detected. In order not to be detected, the attacker chooses the strategy $P = \{p_1, p_2....p_N\}$ where p_i refers to the probability of attacking the target i, the attacker can attack many targets simultaneously. HIDS monitors a set of nodes, continuously as long as the system is running. For the HIDS, it monitors the targets with the probability $q = \{q_1, q_2....q_N\}$, where q_i refers to the probability of monitoring the target i. Monitoring resource has constraints represented as $\sum_{\substack{i=0 \\ i \in S(IDS)}}^{N} q_i < Q$

We assume to start with some reputation budget (an amount of reputation for each HIDS). Reputation assets of different HIDS are independent. For the HIDS, we start with their reputation calculated in the first step. A loss of reputation is represented by -r whose value is proportional to a degree of damage. We suppose that the attacker gains when it is not detected and looses otherwise. The defender can gain reputation while detecting the attacker and can loose otherwise. We establish a game model for the interactions between an HIDS and an attacker. Assuming that both players are mutually aware of the strategies and the utility function they have.

We use the following notations :

- $\boldsymbol{I}$: $|S(HIDS)|$, the number of HIDS nodes
- $\boldsymbol{Z}$: $|S(A)|$, the number of attackers
- $\boldsymbol{r}$: reputation gain
- C_a : cost of attacking, $0 < C_a < 1$,incentive to attack
- C_m : cost of monitoring, $0 < C_m < 1$, incentive to monitor
- C_s : cost of sending false alarm
- $U_a(t)$: the utility functions of the attacker
- $U_d(t)$: the utility function of the defender
- d_i: the detection rate of the node i, $0 < d_i < 1$.
- **(1-** d_i**)**: the false alarm rate of the HIDS.
  The false alarm rate of the HIDS and detection rate are private information to the defender.

Table 5.3: Player payoff

	Monitor	Not monitor
Attack	(1-2d_i) − C_a,-(1-2d_i)r-$C_m r$	r-C_ar, -r
Not attack	0,-$(1-d_i)(C_s r - C_m r)$	0,0

$$U_d(p,q) = \sum_{\substack{i=0 \\ i \in S(IDS)}}^{I} p_i.q_i.(-r_i(1-2.d_i) - C_m.r_i) - p_i(1-q_i).r_i + (1-p_i).(1-d_i)C_s.r_i.q_i + C_m.r_i$$

$$= \sum_{\substack{i=0 \\ i \in S(IDS)}}^{I} r_i.q_i.(p_i(2.d_i + C_s(1-d_i))) - (1-d_i)C_s + C_m \; - \sum_{\substack{i=0 \\ i \in S(IDS)}}^{I} p_i r_i$$

Table 5.3 illustrates the payoff matrix of the attacker defender interaction on target i in the strategic form. The cost of attacking and monitoring are taken into account in our game and are proportional to the reputation of i, denoted by C_a and $C_m r$. $C_s r$ denotes the loss of the false alarms. The cost of attack is less than 1 to make the attacker adopt the strategy of attacking.

Let us start with the utility function of the defender, U_d. Using the payoff table and the strategies of the players, we can define U_d as :

Now, we can also define the utility function of the attacker, U_a as: Before solving the game, we can resume the definition of the network intrusion detection with one attacker and one defender as follows:

Players : Attacker, Defender (HIDS)

Strategy : Attacker : p

Defender: q / $\sum_{\substack{i=0 \\ i \in S(IDS)}}^{N} q_i < Q$

Payoff: $U_a(p,q)$ for the attacker and $U_d(p,q)$ for the defender.

Game: The attacker and the defender selects its strategy p/q to maximize U_a and U_d.

Game solution This game cannot be solved using pure strategy, so mixed strategy is used to derive our solution. A result established by John Nash in [114] (1951), and captured in the following theorem.

Theorem 2 *Every N-player nonzero-sum game has a Nash equilibrium in mixed strategies.*

Definition 16 *A strategy profile (q^*, p^*) is said to be an Nash Equilibrium, NE of the network if neither the attacker nor the defender can improve its utility.*

We consider that the attacker has a strategy p. The attacker can attack one target at least with non zero probability. We consider case where there are only one attacker and one defender in the following to analyze the optimal strategy.

Defender wants to maximize the minimum payoff. Therefore, the leader will calculate the first derivative of U_d with respect to the variable q_i. The purpose of the leader is to have the value p^* to decide whether to adopt monitoring.

$$U_a(p,q) = \sum_{\substack{i=0 \\ i\in S(A)}}^{Z} p_i.q_i.((1-2.d_i)-C_a.) + p_i(1-q_i)(r_i-C_a))$$

$$= \sum_{\substack{i=0 \\ i\in S(A)}}^{Z} p_i(1-2.d_i p_i - C_a)$$

Table 5.4: Simulation testbed

Scenario 1	$C_s = 0,01$,$C_a = C_m = 0,001$
Scenario 2	$C_s = 0,3$,$C_a = C_m = 0,01$

In the leader side: As, we said before, it derivates the utility function of the defender (or HIDS), U_d with respect to q_i using equation the previous utility function, U_d. We replace in U_d p_i by p^* and we derivate with respect to q:

$p^*(2d_i + (1-d_i)C_s) - 1$

We put the equation equal to zero. So, we obtain $p^* = \frac{1}{2d_i+(1-d_i)C_s}$

When $p > p^*$, the leader informs the target node to run its HIDS.

The attacker wants to minimize the expected payoff for the defender player. Therefore, attacker will calculate the first derivation of U_a with respect to the variable q_i to have the q^*. Using this equation $U_a(p,q) = \sum_{\substack{i=0 \\ i\in S(A)}}^{N} p_i(1-2.d_i p_i - C_a)$

We replace q_i by q^* and derivate with respect to p to obtain:

$p^*(1-2d_i q^* - C_a)$

To have the minimum we must have the first derivate equal to zero.

The equilibrium strategy will be : $q^* = \frac{(1-C_a)}{2d_i}$

The equilibrium formula shows that q^* depends on the cost of the attacking. Also, the attacker decides to attack when $q < q^*$.

5.3 Simulation analysis

In this section, we present our testbed. The simulation experiments are implemented using MATLAB. Let us consider a network with 200 nodes and with various number of potential attackers. The selection of nodes strategy is chosen randomly.

We consider a network with a high requirement on trust, e.g., execution on cloud computing normally requires a high level of assurance and needs to be resistant to many types of network attacks. In such scenario, we consider low cost of attacking and monitoring. Defenders must have a large value of detection and then a small false alarm rate. The second scenario we consider is when the cost for monitoring and attack cost are important e.g., a wireless local area network where defender have limited resource. In this case, HIDS is not so performing. The table 5.4 resumes scenario 1 and 2.

We start with one attacker and one defender, the total attack and monitor resource are set to 1. We calculate the correspondent optimal payoff for the defender and

the attacker. We can affirm that with low cost, we have more important values of NE strategy (0,49; 0,56). For the high cost scenario, we obtain (0,47; 0,45). The question was whether a rational attacker will focus on some targets or allocate its attack resource to all targets to reduce the probability of being detected. To explain numerical result and analyze the question, we can say that both the attacker and the HIDS focus only on sensible targets with low trust.

For scenario 1, the decision of the leader is to let at least 80% of the HIDS running simultaneously at the probability 0.533. The minimum number of attacking risks seen by the leader is one attack against one target.

For scenario 2, the decision of the leader is to let at least 99% of the HIDS running simultaneously at the probability 0.472. The minimum number of attacking risks seen by the leader is one attack against one target. This result confirms that the decision depends on the parameters such as d, (1-d) and costs.

We aim now to show the improvement brought by game theory in this framework. For this, we consider the resource consumption in two cases:

- first, considering only the first phase with all monitoring' resource used in the cooperative detection.
- second, considering the complete framework with allocation of the resource based on the equilibrium of the game.

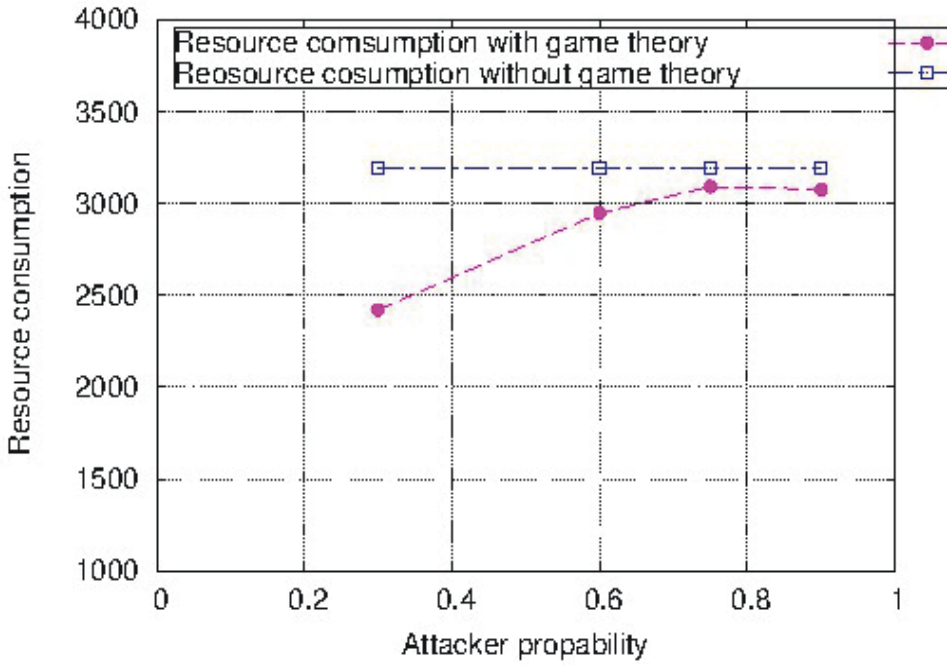

Figure 5.2: Resource consumption

We obtained the result shown in figure 5.2. This figure depicts that the gap between the two approaches (curves) is very high especially in the beginning. It can

be seen that about 25% gain is achieved between 0,2 and 0,6 (real case). Thus, our combined solution provides an excellent optimization of the resources.

Another important parameter in this work is the detection rate. We suppose that we know the strategy of the attackers. We can evaluate the detection rate by comparing the evaluation of the HIDS and what we know. Detection rate is calculated in each HIDS as the positive evaluation.

As shown in the figure 5.3, with game theory the detection rate is better than without considering this game-theoretic phase. Therefore, this game notifies victims of possible attacks to launch their detection mechanisms. This will ameliorate the detection as we are sure that system is up and can analyze the misbehavior.

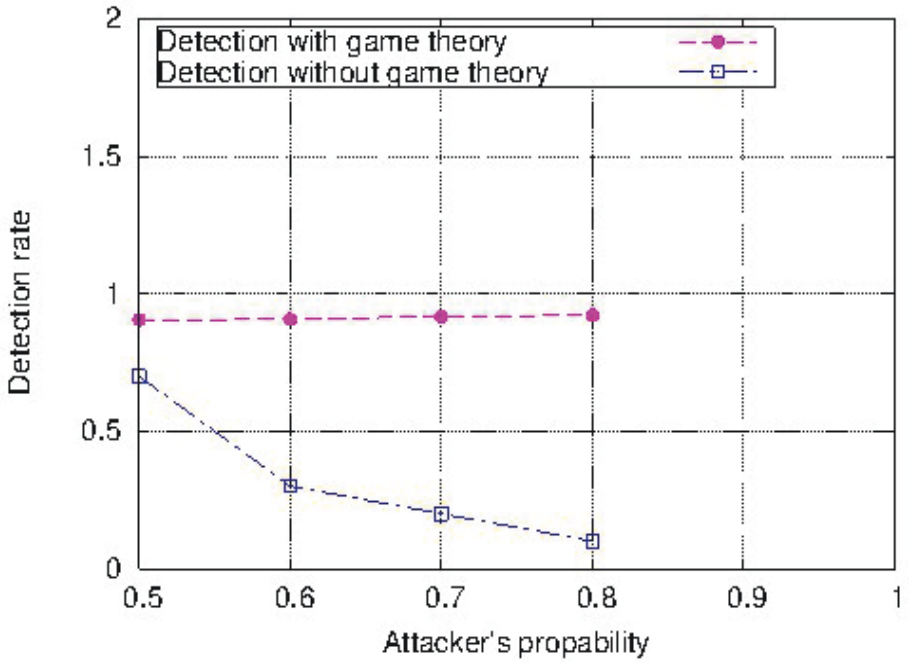

Figure 5.3: Amelioration of detection

5.4 Conclusion

In this chapter, we designed a new framework with game theory and election process to improve detection of malicious nodes, optimize resources and cope with heterogenous domains. Here, one actor is the IDS and the second is the malicious. The IDS tries to improve detection and the malicious tries to compromise. We show that trust and reputation are major issues when we have various participants. Our game-based framework, with trust management and efficient election, can improve the security and reduce resource consumption and monitoring management.

In the next chapter, we transform our system into a distributed gossip based trust system and we study the behaviors of stubborn peers.

Chapter 6

Gossip-based reputation systems and stubborn Peers

Contents

6.1 Introduction

The open nature of networks makes the system vulnerable to malicious peers trying to provide misleading services. Consequently, a major challenge for distributed systems is how to establish trust between different peers without the benefit of trusted third parties or authorities.

In chapter two and three, we assumed that we had a central entity. This central entity aggregated all the trust and estimated the reputations. In this part, we want to have a distributed system to allow peers to manage the trust and reputation. However, malicious peers can appear in this distributed reputation management and act as stubborn and forceful peers. So, a good method for trust aggregation need to be designed for an effective reputation management system. This chapter is divided into three parts. First, we describe a gossip based reputation management in the normal case (let us assume that a normal case is when peers are respecting the model) and the case of stubborn peers (some peers don't respect the model). Second, we implement and validate our proposed mathematical proposal with an illustrated example. Finally, we analyze the spread of misinformation and persistent disagreement in case of stubborn behaviors.

Table 6.1: Definitions

Notation	Definition
d^t_{vw}	The local reputation score of peer w given by peer v at the time t
R^v	Global reputation vector for peer v
numOfPeers	Number of peers
deg(v)	Number of neighbor of peer v
α	The weight of the local evaluation

6.2 Aggregation and gossip-based protocol

Modern distributed systems often use gossip protocols to solve problems that might be difficult to solve [121] [116]. Gossip-based protocols, as a style of communication protocols, is appealing for large-scale distributed applications such as information dissemination, aggregation, and overlay topology management. The roots of gossip-based approach can be traced to a paper by Frieze and Grimmett [121]. Gossip algorithm can support any aggregation process [122]. However, in the literature, not many studies use the gossip algorithm to build trust management system. In [116], authors proposed an efficient method for aggregation of global reputation values in peer-to-peer networks.

Gossip algorithms are used for information spreading in large distribute networks. In these algorithms, nodes randomly choose their communication partner in each information diffusion step. These algorithms are generally simple, light weight and robust for errors. They have less overhead compared to deterministic algorithms [118], [117]. Gossip algorithms are also used for one distributed computation like taking of averages. This type of algorithms is also used for computation of reputation vector in peer-to-peer networks [116]. There are three types of gossip algorithms: push, pull and push-pull. In push kind of algorithms, in every gossip step nodes randomly choose a node among their neighbors and push their information to it. Whereas, in pull algorithms, nodes take the information from the randomly selected neighboring nodes. Both these processes happen simultaneously in the push-pull based algorithms.

Authors in [120] studied the reaction of the GossipTrust system [116] to group empowering. In others words, they consider malicious peers in single group and in pairs. They evaluated the percentage of malicious peers that do not affect the normal behavior of the system.

6.3 Proposed gossip-based reputation management

Our distributed management model considers local ratings and recommendation coming from other peers. In our system, a peer makes decisions based on its experience and other peer's recommendation. The challenging problem is how to aggregate values from different recommenders in an efficient manner. In the table 6.1, we present all the notations and parameters used in this work.

Definition 17 d^t_{vw}, *the evaluation between two peers, is the local observation of peer* v *for peer* w *during the period of time* t. *The evaluation between two peers can be*

estimated based on local observations of their behaviors. d^t_{vw} *reflects how consistent two nodes are and therefore how trustworthy they think each other are. The more frequently two peers give inconsistent results, the less evaluation score between them becomes. In our model, after a direct interaction between the two peers, evaluator will evaluate neighbor according to the quality of storage provided by its neighbor and then locally store the results. If there is no evaluation between two peers,* d^t_{vw} *is set to zero. Time index will be skipped in the remaining of this chapter.*

Definition 18 R^v *is a column vector of size* $(numOfPeers - 1)$ *representing the global reputation of the peer v seen by other peers (after convergence).*

The goal of our framework is to estimate the average of trust spread over a distributed network. Our gossip based algorithm consists of 2 steps : the local reputation recommendation and the global trust evaluation. Let peers in the networks are represented by a set P, their ability to communicate with each other is represented by a set of edges. The fact that peer v and peer w are able to communicate is conditioned by the euclidean distance between these two peers and the configured range. The communication is represented by $v \sim w$ else $v \nsim w$. In the Fig.6.1, this communication is represented by a blue line between peers. Let us suppose that this graph is 2-vertex connected. Each peer holds information as a rating value about the others. More precisely, every peer maintains a reputation vector containing all ratings about other peers. We suppose that peers interact with others and update their information dynamically taking into account implications of this information both today and in the future. We represent this information as a vector of reputation, R, meaning that peer v holds the scalar R_{vw} about w. Two peers are allowed to communicate only if they are connected in the network. Obviously, there are many ways to satisfy this constraint: peers can communicate pairwise or not, synchronously or not, etc...

Now, each peer has its own initial information and wants to share it with the other peers so that any one can estimate the global vector R. When two peers v and w are connected in the network, the initial information held is d_{vw}.

Considering the connections between peers, we can explain the update of R^v as following:

Algorithm 10: Update process of R^v

1: If $w \sim v$ then
2: $R_{wv} \leftarrow \alpha.d_{wv} + \frac{(1-\alpha)}{deg(w)-1} \sum_{k\sim w, k\neq v} R_{kv}$
3: If $w \nsim v$ then
4: $R_{wv} \leftarrow \frac{1}{deg(w)} \sum_{k\sim w, k\neq v} R_{kv}$

Since the graph is 2-vertex connected, $deg(w) >= 2 \ \forall$ w. So, in the previous algorithm, we can divide by $deg(w) - 1$.

6.3.1 Case of normal peers

In this part, we want to analyze the previous proposed model in case all peers are cooperative and respect the update process. If we fix a node and we want to observe

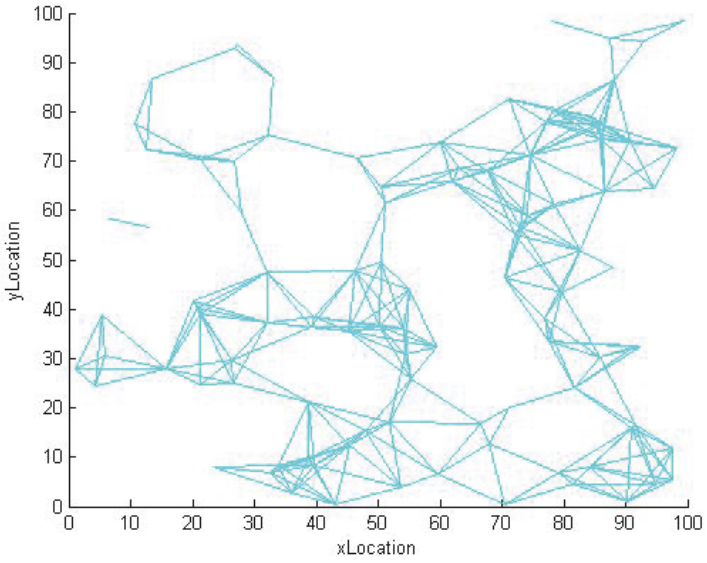

Figure 6.1: An example of the network of 100 peers

the global reputation, we can write the column corresponding to this node as follows: $R^v = A^v R^v + h$, where $h = \alpha * d$

A^v is the a square matrix of $(numOfPeers - 1)$, where in the row w we have $\frac{(1-\alpha)}{deg(i)-1}$ for the neighbors of v (if $i \sim v$) and $\frac{(1)}{deg(i)}$ for the non neighbor of v (if $i \nsim v$).

Theorem 3 *The global reputation vector R^v_∞ converges to $(I - A^v)^{-1} * h$ where I is the identity matrix.*

Proof:

We start with the equation $R^{v,n} = A^v R^{v,n-1} + h$. Repeating the argument, we get $R^{v,n} = A^{v,n} R^{v,0} + (A^{v,n-1} + A^{v,n-2} ... + A^v + I) * h$

We have $\rho(A^v) < 1$ (ρ is the spectral radius). From $\rho(A^v) < 1$ we deduce that A^v tends to 0 and $(A^{v,n-1} + A^{v,n-2} ... + A^v + I) * h$ tends to $(I - A^v)^{-1} * h$. For each complex number z with $|z| > 1$, let us prove that matrix $(zI - A^v)$ is invertible.

Assume that $(zI - A^v)$ be not invertible, then there exists a vector $x \neq 0$ such that $A^v x = zx$. Let us now denote a node as v0. We have, $zx(v0) = \sum_w A^v(v0, w)x(w)$. Hence, $|z||x(v0)| <= \sum_w A^v(v0, w)|x(w)|$. By definition of v0:$|z||x(v0)| <= \sum_v A^v(v0, v))|x(v)|$. We have $\sum_v A^v(v0, v) <= 1$. Hence, necessarily, $\sum_w A^v(v0, w) = 1$. Moreover, $A^v(v0, w) > 0$ implies that $x(w) = zx(v0)$, otherwise the equality $zx(v0) = \sum_w A^v(v0, w)x(w)$ would be violated. So one can repeat the argument with all the neighbors of v0: all their own neighbors are necessarily in R and eventually all the connected component containing v0 lies in R, which completes the proof.

■

6.3.2 Case of stubborn peers

In this section, we study the previous proposal of gossip based reputation system under the assumption that some peers misbehave. We show how results can be wrong under this assumption.

We assume that some malicious peers, called stubborn, never change their evaluation regarding other peers.

The rationale behind this is twofold: either they are malfunctioning, they might be malicious and influence the network, or simply do not respect the protocol and optimize resources. Stubborn peer is driving the network to a different state instead of the sought average state when using standard gossip algorithms.

Suppose that C is the set of cooperative peers and NC is the set of non cooperative peers where both sets are assumed to be non empty. We study this case of network assuming:

Assumption 1:

Vertices is the disjoint union of cooperative peers C and non cooperative stubborn peers. Both sets are assumed non empty. We can then rewrite the vector R^v as:

$$R^v = \begin{bmatrix} R^v_C \\ R^v_{NC} \end{bmatrix}$$

Suppose that the network is the union of regular cooperative peers and non-cooperative peers (malicious). We can write the network N as following:

$$N = \begin{bmatrix} C \\ NC \end{bmatrix}$$

Note that, by assumption 1, the state vector of stubborn peers, R^v_{NC}, is constant over time.

Assumption 2:

1/ A^v is a right stochastic matrix: A^v has its entries in $[0, 1]$ and $A^v 1 = 1$,

2/ A^v is written in the following bloc form.

$$A^v = \begin{bmatrix} A^v_C & A^v_{NC} \\ 0 & I \end{bmatrix}$$

We now address the issue of convergence of gossip-based model in the presence of stubborn peers. Let us begin with a lemma.

Lemma 1 *Under the assumption 2, for each number z with $|z| >= 1$, matrix $zI - A^v_C$ is invertible. Specifically $I - A^v_C$ for $z = 1$*

Theorem 4 *Under the assumption 1 and 2, $R^{v,n+1} = A^{v,n} R^{v,n}$ converges to $R^v_\infty = (I - A^v_C)^{-1}(h + A^v_{NC} * R^v_{NC})$*

Consensus is not necessarily reached since R^v_∞ is not proportional to 1 as long as stubborn peers disagree with each other. In addition, the limit state does not depend on the initial state, it only depends on the stubborn peers state and the h vector. In other terms, the stubborn peers drive the network, initial opinions of normal peers is lost, even with one single stubborn peer.

For the proof of the convergence, it is the same as for the previous theorem in the case of normal peers.

6.3.3 Validation of the convergence and example

6.3.3.1 Validation

In this section, we validate the theoretical convergence shown previously. For this, we implement the process of gossip in Matlab. We simulate a graph where the fact that a peer v and w are able to communicate is when the euclidean distance between two peers is less than the transmission range, so they are considered as neighbors. Each peer in our simulation can update the reputation of other peers. Alg 11 represents the global process to calculate the reputation of one peer. In this algorithm, we represent how a peer can build the reputation and how the global reputation vector converges. This convergence is conditioned by the small value of ϵ. In Alg 11, we distinguish between two algorithms: one for the update of reputation between neighbors $LocalUpdate$.

Algorithm 11: Global peer reputation

Require: numOfPeers
Ensure: R
1: Initialize the global reputation matrix, R
2: **repeat**
3: $T \leftarrow R$
4: **for** $i \in 1..numOfPeers$ **do**
5: **for** $j \in 1..numOfPeers$ **do**
6: Local Update (R_{ij})
7: **end for**
8: **end for**
9: **until** $T - R < \epsilon$

To explain more the process of reputation aggregation, we present Alg 12. This algorithm represents the function Local Update used in the previous algorithm.

Algorithm 12: Local Update

Ensure: R_{ij}
Require: i, j, G, numOfPeers
1: **for** $v \in 1..numOfPeers$ **do**
2: **if** $(i \sim v)$and $v \neq j$ **then**
3: $G = G + R_{vj}$
4: **end if**
5: **end for**
6: **if** $(i \sim j)$ **then**
7: $R_{ij} \leftarrow \alpha.d_{ij} + \frac{(1-\alpha)}{deg(i)-1}G$
8: **else**
9: $R_{ij} \leftarrow \frac{1}{deg(i)-1}G$
10: **end if**

In the Alg. 12, a peer i, neighbor of j, takes its local evaluation about the peer j and the recommendation of its neighbors. In fact, to aggregate the recommendation

of other peers, peer i calculates the sum of all the recommendations of reputation and divide them by the number of the real neighbors (not considering the evaluation of j for itself).

A peer i, not neighbor of j, takes only the recommendation of its neighbors about peer j. For aggregation, peer i calculates the sum of all the recommendations of reputation and divide them by the number of the real neighbors.

We implement R_∞ as shown in the previous section to compare and verify the results. Note that $R_\infty = (I - A)^{-1} * h$. For that, we must compute A^v for each peer v. The process will be shown in the Alg. 13. After A^v for each peer, we continue the computation of $(I - A)^{-1} * h$. For more details, an example of this matrix is provided in the next section.

Algorithm 13: Construction of A

Ensure: A_v
Require: v, numOfPeers, deg, α
for $i \in 1..numOfPeers$ **do**
  if "$(i \sim v)$and $i \neq v$" **then**
    $H = \alpha * d_{iv}$
    for $w \in 1..numOfPeers$ **do**
      if "$(w \sim i)$and $w \neq i$ and $w \neq v$" **then**
        $A_{iw} = ((1 - \alpha)/(deg(i) - 1))$
      end if
    end for
  else
    $H_{vi} = 0$
    for $w \in 1..numOfPeers$ **do**
      if "$(w \sim i)$and $w \neq i$ and $w \neq v$" **then**
        $A_{iw} = (1/deg(i))$
      end if
    end for
  end if
end for

We implement the Alg. 11 in the graph in Fig. 6.2. By comparing the results, we obtain the same vector R_∞. So, two different methods can lead to the same convergence and that justify our theoretical model.

6.3.3.2 An example

The implemented process is best illustrated by a small example of connected peers in the figure 6.2. Consider a P2P connected network with six peers. At the time $t_0 = 0$ the global reputation vector is initialized. Let us also consider $\alpha = 0.3$ for each peer. The topology is important to determine the way to update the reputation vector in the peers. The updated reputation vector is calculated as shown in the Alg. 11. Let us simulate the update of the reputation vector of P5.

Table 6.2: neighborhood table

1	1	0	1	0	0
1	1	0	1	1	0
0	0	1	0	0	1
0	1	0	1	1	0
0	1	0	1	1	1
0	0	1	0	1	1

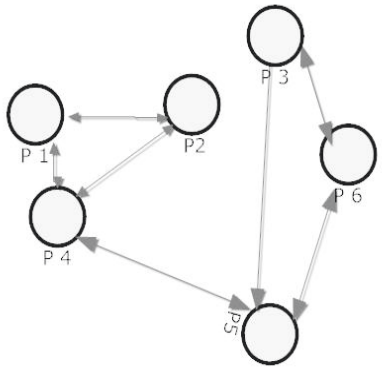

Figure 6.2: Topology of the example network

Table 6.3: Reputation aggregation process for P5

Gossip Step	Peer 1	Peer 2	Peer 3	Peer 4	Peer 5	Peer 6
0	0	0.9838	0	0.32	1	0.9168
Way to update	$(R_{25} + R_{45})/2$	$0.3 * 0.9838 + 0.7/2$ $*(R_{15} + R_{45})$	R_{65}	$0.3 * 0.32 + 0.7/2$ $*(R_{25} + R_{15})$	1	R_{65}
1	0.6519	0.6353	0.9168	0.5406	1	0.9168

The table 6.2 represents the communication in Fig.6.2 (where 1 means neighbors). We can affirm that P5 and P2 are neighbors, P5 and P6 are neighbors and P5 and P4 are neighbors. We have the vector d that can evaluate the opinion about neighbors locally. Initially the reputation vector of P5 is (0, 0.9838, 0 ,0.32, 1, 0.9168).

Table 6.3 is used to illustrate the procedure and update in Fig. 6.2. At each gossip step, every peer executes two procedures: One sends the local reputation to the neighbor. Another receives the reputation recommendations from other peers and computes the updated the reputation as explained before.

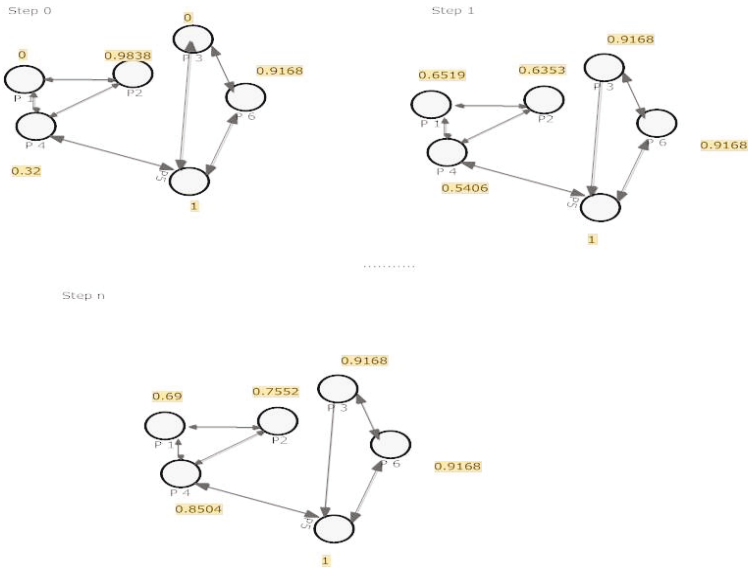

Figure 6.3: Evolution of P5 reputation

Initially at step 0, we thus assume that only local reputation are considered. At the first gossip step, P2 sends the value (0.9838) to P4 and P1. In the same time, P4 sends the value (0.32) to P1 and P2 and P6 sends the value (0,9168) to P3. Then P2 and P4 updated reputation value as explained in the table 6.3. After the first gossip step update, P1 has the reputation value 0.6519 about P5, P2 has 0.6353 and P4 has 0.5401.

In the Fig 6.3, we resume the reputation obtained by P5 in step 0, step 1 and

Table 6.4: A^5

0	0.1	0	0	0	0
0.0778	0	0	0.0778	0	0
0	0	0	0	0	0.1667
0.1667	0.1667	0	0	0	0
0	0	0	0	0	0
0	0	0.1	0	0	0

Table 6.5: Impact of stubborn peer P2 on the reputation of P5

d(P2)	Peer 1	Peer 2	Peer 3	Peer 4	Peer 5	Peer 6
0.2	0.2	0.4286	0.3139	0.3026	1	0.255
0.6	0.6	0.6236	0.3643	0.4581	1	0.3271

step n. After a certain number of steps and following the same process of update, we have reached the convergence of the reputation vector of P5. Suppose we extend the gossiping process to another step after the convergence, we will see no more changes in the values. Without centralized control, the consensus must be determined on distributed nodes locally in each peers.

We can adopt the second method to have the final reputation vector $R^5_\infty = (I - A^5)^{-1} * h$.

Based on the neighbors table 6.2, we compute the square matrix A^5. The table 6.4 illustrates this matrix.

6.3.3.3 Stubborn peers and the persistent disagreements

As we said, there are two types of peers : normal and stubborn. Normal peers exchange information with their neighbors and modify them. In contrast, stubborn peers never update their local reputation and continuously influence others. We show that the presence of these peers leads to persistent disagreements.

We suppose that there is a stubborn peer P2 $\in NC$. P2 tries to influence the reputation by including the same value in the update process. We will take for this example two possible values (0.2 and 0.6).

We take the example of six connected peers with P2 as stubborn peer as shown in the Fig 6.4. Suppose that P2 evaluates its neighbors with the same values and do not change: for the first simulation, the value is 0.2, for the second one, the value is 0.6. In the table 6.5, reputations of the peer 5 are different and depend on the value given by the stubborn peer P2. For every value of the stubborn peer, there exists a converged values. This result confirms the impact of stubborn peers even one peer can change the evaluations.

Results also show that for the reputation given by the stubborn peer affects the convergence. That confirms the formula obtained for the final global reputation $(I - A^5_C)^{-1}(h + A^5_{NC} * R^5_{NC})$. In this formula, we affirm that the convergence depend on the stubborn peers. From the results, it is shown that using gossip algorithms in the presence of stubborn peers leads to non-desirable results.

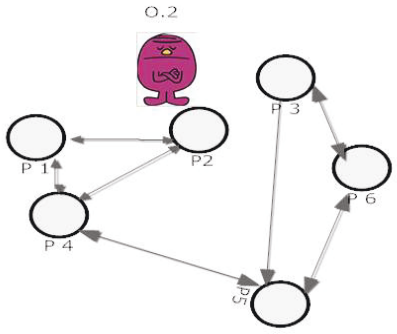

Figure 6.4: Topology of the example network with stubborn peer

6.4 Conclusion

In this chapter, we build a gossip based model to aggregate reputation in fully distributed system. Our focus in this chapter has been on models linking the dynamics of reputation to the propagation of local reputation, the form of updating and crucially the behaviors of peers. In particular, we highlighted the importance of these factors on two results:

- Our model lead different individuals, who might start with different local reputation and engage in communication with different subsets of the network, to hold global reputation.
- Misinformation: stubborn peer are able to manipulate the reputation of others.

In the following contributions, we will focus on the forceful peers and their strategies. This kind of peers can also affect the process of reputations.

Chapter 7

Simulations of forceful peers in gossip-based reputation systems

Contents

7.1 Introduction

Forceful peers influence others disproportionately in reputation management for distributed systems. We can interpret forceful peers in multiple different ways. First, forceful peers may correspond to community leaders providers, who have a disproportionate effect on the opinions of their neighbors. Second, forceful peers can be the media power. Our purpose in this part is to determine whether those media sources, prominent agents, politicians and the state will be able to manipulate beliefs and spread misinformation in a the network. We are motivated by the political voting system and the impact of media resource to change opinions. An example of the obtained results in this system is shown in the Fig.7.1. We can show that one part

has the majority of the voting. This affirms the efficiency of his strategy to impact the opinions.

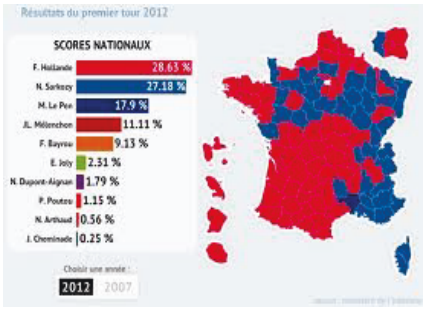

Figure 7.1: Voting system

In this chapter, we describe our distribute reputation management and strategies that can be adopted by forceful peers. We aim to identify the best strategy that employs those peers to manipulate the network and alters the reputations. We test the strategies under different graphs to open a large choice of networks types. In this

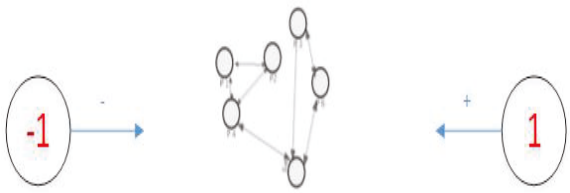

Figure 7.2: The network, an example

chapter, we take the example of reputation update to show the impact of forceful peers. We test different strategies that can be used by those peers under different communication graphs. Let us suppose that the network is the union of regular cooperative peers, C, and two forceful peers (Fig.7.2). We can write the network N as following:

$$N = \begin{bmatrix} F \\ C \end{bmatrix}$$

Where

$$F = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

Two forceful peers affect the reputation model: one has the positive (1) influence and an other acts negatively (-1). In the network, vertices are the disjoint union of

Table 7.1: Definitions

Notation	Definition
d_i	The local reputation of peer i
R	Global reputation
BMax	The maximum value of budget (amount of points to distribute)
Re	Remaining resource after each affectation
deg(i)	The degree of the peer i
numOfPeers	Number of peers
α	Weight of d_i
Aff_-	Affectation of peer (-1)
$nAff_+$	Number of peers affected by peer (1)
Aff_+	Affectation of peer (1)
$nAff_-$	Number of peers affected by peer (-1)

cooperative peers C and forceful peers F. We can then rewrite the vector R in bloc

$$R = \begin{bmatrix} R_C \\ R_F \end{bmatrix}$$

In the following, an overview of the proposed reputation management explained in the previous chapter, gossip-based reputation aggregation, is represented in case of forceful peers. In summary, the strategies build by forceful peers. Then, we simulate matches between two forceful peers with different strategy and on different graph. Results are shown and analyzed to make conclusions for each graph.

7.2 Reputation management overview

The goal of our framework is to estimate the average of reputation spread over a P2P network. For this, we work with the same model of aggregation in the previous contribution, gossip-based reputation aggregation. More notations are explained in the table 7.1. We will focus on the local evaluation and the impact of forceful peers on the recommendation. For that, we have a simple vector of reputation, R as we mention in the Alg. 14.

Alg 15 represents the function LocalUpdate used in Alg. 14. $nAff_-, Aff_-, nAff_+$ and Aff_+ are used in the Alg 15 to represent the impact of the the peer (-1) and the peer (1). The vectors Aff_andAff_+ are obtained after the execution of the strategies that are explained in the next section. In Alg 15, we have the influence of the forceful peers represented as $nAff_-$ and $nAff_+$. This impact is simulated exactly by a number of points affected to a chosen peers (virtual link). For the forceful peer (-1), the affectation is negative. For the forceful peer (1), the affectation is positive. The convergence of the update process is assured as explained in the previous chapter (we suppose that we have two stubborn peers). $R^{v,n+1} = A^{v,n}R^{v,n}$ converges to $R^v_\infty = (I - A^v_C)^{-1}(h + A^v_F * R^v_F)$, where

$$F = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

Algorithm 14: Reputation aggregation

Require: numOfPeers
Ensure: R
Initialize R
repeat
  T $\leftarrow R$
for $i \in 1..numOfPeers$ **do**
    LocalUpdate (R(i))
end for
until $T - R < \epsilon$

Algorithm 15: LocalUpdate

Ensure: R
Require: i, j ,s,$nAff_-, nAff_+, Aff_-, Aff_+$
for $v \in 1..numOfPeers$ **do**
  if "$(i \sim v)$" **then**
    s=s+R(v)
  end if
end for
$R_{ij} \leftarrow \alpha.d_{ij}^t + \frac{(1-\alpha)}{deg(i)+nAff_-+nAff_+}(s + Aff_- + Aff_+)$

We implement the proposed algorithms in Matlab and validate the convergence as we have done in the previous chapter. Validation results confirm the theoretical aspect of our model.

7.3 Description of possible forceful peers strategies

There is the possibility that some peers are prominent and may influence others. This enables us to model some peers trying to manipulate the reputation of others for spreading their opinions. Forceful peers adopt many strategies to influence the network and change the general reputation of peers. Four strategies are proposed and illustrated in this section. We choose to test different possible strategies. Our proposed strategies are based on parameters related to the graph concept (especially the peers degree). We prefer to handle with this parameter due to our proposed reputation model which is based on communication between peers.

7.3.1 Strategy U: Random uniform

One forceful peer (1 or -1) can choose the random uniform strategy shown in Alg.16. It is random because, a forceful peer builds the list of affected peers randomly. So, the strategy consists on generating a random number that will be the number of the affected peer. After that, the forceful peer affects this random peer, updates the

vector of affectation and decreases the budget. This process is repeated until the exhaustion of the budget.

Algorithm 16: Strategy U: Random uniform

Ensure: Affectation for each peer, $Aff_* : Aff\_or Aff_+$
Require: numOfPeers
1: **while** $R < BMax$ **do**
2: Generate a random number x in between 1 and numOfPeers
3: $Aff_*(x) = Aff_*(x) + 1$
4: $nAff_* = +1$
5: Re=Re-1
6: **end while**

7.3.2 Strategy D: Degree

One forceful peer can choose the strategy "degree" shown in Alg.17. It is called "D" because the affectation depends on the degree of the peer. For that, the algorithm calculates for each peer the fraction $P_i = \frac{deg(i)}{sum(deg)}$ and builds all intervals $[P_i, P_i + P_{i+1}]$ for all peers i. This process will privilege the peers having more degree for the affection. As in the previous strategy, the algorithm generates a random number, checks the belonging of the number to one of the intervals $[P_i, P_i + P_{i+1}]$, affects the peer and decreases the budget. This process is repeated until the exhaustion of the budget.

Algorithm 17: Strategy D: Degree

Ensure: Affectation for each peer, $Aff_* : Aff\_or Aff_+$
Require: numOfPeers
1: **for** $i \in 1..numOfPeers$ **do**
2: Calculate $P_i = \frac{deg(i)}{sum(deg)}$
3: **end for**
4: Determine the intervals $[P_i, P_i + P_{i+1}]$
5: **while** $R < BMax$ **do**
6: Generate a random real number x between 0 and 1
7: Determine i verifying $x \in [P_i, P_i + P_{i+1}]$
8: $Aff_*(i) = Aff_*(i) + 1$
9: $nAff_* = +1$
10: Re=Re-1
11: **end while**

7.3.3 Strategy 1/D: 1/degree

One forceful peer can choose the strategy "1/degree" shown in Alg.18. It is called "1/D" because the affectation depends on the inverse of degree. For that, the algorithm calculates for each peer the fraction $P_i = \frac{1/deg(i)+1}{sum(1/deg+1)}$ and builds all intervals

$[P_i, P_i + P_{i+1}]$ for all peers i. This process will privilege the peers having more degree for the affection. As in the previous strategy, the algorithm generates a random number, verifies the belonging of the number to one of the intervals $[P_i, P_i + P_{i+1}]$, affects the peer and decreases the budget. This process is repeated until the exhaustion of the budget.

Algorithm 18: Strategy 1/D : 1/Degree

Ensure: Affectation for each peer, $Aff_* : Aff\_ or Aff_+$
Require: numOfPeers
1: **for** $i \in 1..numOfPeers$ **do**
2: Calculate $P_i = \frac{1/deg(i)+1}{sum(1/deg+1)}$
3: **end for**
4: Determine the intervals $[P_i, P_i + P_{i+1}]$
5: **while** $R < BMax$ **do**
6: Generate a random number x between 0 and 1
7: Determine i verifying $x \in [P_i, P_i + P_{i+1}]$
8: $Aff_*(i) = Aff_*(i) + 1$
9: $nAff_* = +1$
10: Re=Re-1
11: **end while**

In $P_i = \frac{1/deg(i)+1}{sum(1/deg+1)}$, we put +1 to avoid dividing by 0 if the nodes haven't neighbors.

7.3.4 Strategy D^2: Square degree

One forceful peer can choose the strategy "$Degree^2$" shown in Alg.19. It is called "D^2" because the affectation depends on the inverse of degree. For that, the algorithm calculates for each peer the fraction $P_i = \frac{deg(i)^2}{sum(deg^2)}$ and builds all intervals $[P_i, P_i + P_{i+1}]$ for all peers i. This process will privilege the peers having more degree for the affection. As in the previous strategy, the algorithm generates a random number, checks the belonging of the number to one of the intervals $[P_i, P_i + P_{i+1}]$, affects the peer and decreases the budget. This process is repeated until the exhaustion of the budget.

7.4 Results and analysis on each graph type

To estimate the efficiency of the forceful strategies, we consider a battle between two peers. We choose to test the strategies under two different amount of budget, BMax, high budget, 200 and low budget, 50. The same matches are simulated on different type of graph : geometric graph, random erdos Renyi graph and scale free graph. Suppose in the system, peers are cooperating to evaluate an object or a new peer in the system. Suppose also that two peers are forceful: one peer wants to impact negatively and an other positively. The percentage shown in the results in this chapter corresponds on the percentage of peers that evaluate others negatively

Algorithm 19: Strategy D^2: $Degree^2$

Ensure: Affectation for each peer, $Aff_* : Aff\_ or Aff_+$
Require: numOfPeers
1: **for** $i \in 1..numOfPeers$ **do**
2: Calculate $P_i = \frac{deg(i)^2}{sum(deg^2)}$
3: **end for**
4: Determine the intervals $[P_i, P_i + P_{i+1}]$
5: **while** $R < BMax$ **do**
6: Generate a random number x between 0 and 1
7: Determine i verifying $x \in [P_i, P_i + P_{i+1}]$
8: $Aff_*(i) = Aff_*(i) + 1$
9: $nAff_* = +1$
10: Re=Re-1
11: **end while**

or positively. Results are obtained after 25000 simulations of the match between each pair of strategies. In the results, as we have some cases with equal percentage of dominance, so, the sum of dominance of both strategy (in battle) is not equal to 100%.

7.4.1 Geometric graph

In the geometric graph, the fact that a peer v and w are able to communicate is when the euclidean distance between them is less than the transmission range. In this case, there are considered as neighbors. This graph can be called unit disk graph as shown in Fig.7.3. In graph theory, a unit disk graph is armed by the intersection of unit disks in the euclidean plane. That is, we form a center of disk as vertex, and connect two vertices by an edge whenever the corresponding disks have non-empty intersection.

7.4.1.1 Uniform strategy in geometric graph

In this section, an evaluation of the random uniform strategy, U, is presented. To reach this purpose, we simulate matches between uniform strategy and the other strategies, degree, $1/degree$ and $degree^2$. Suppose that one forceful peer is choosing the uniform strategy to impact the peers in the geometric graph positively, an other forceful peer (acting against as negative impact) chooses strategy D or $1/D$ or D^2 (D= degree, $1/D = 1/degree$ and D^2=$degree^2$). We estimate the percentage of peers that evaluate others negatively (influenced by strategy D or 1/D or D^2) or positively (so influenced by U). To have good analysis, we investigate in many different simulations and a very big number of simulation. In our case, the results are "to win" or "to loose", (X between [0..1]). The standard deviation is $\sqrt{E(X^2 - E(X)^2}$ Our results are obtained after 25000 simulations of each match between two strategies in four cases:

- the same high budget,

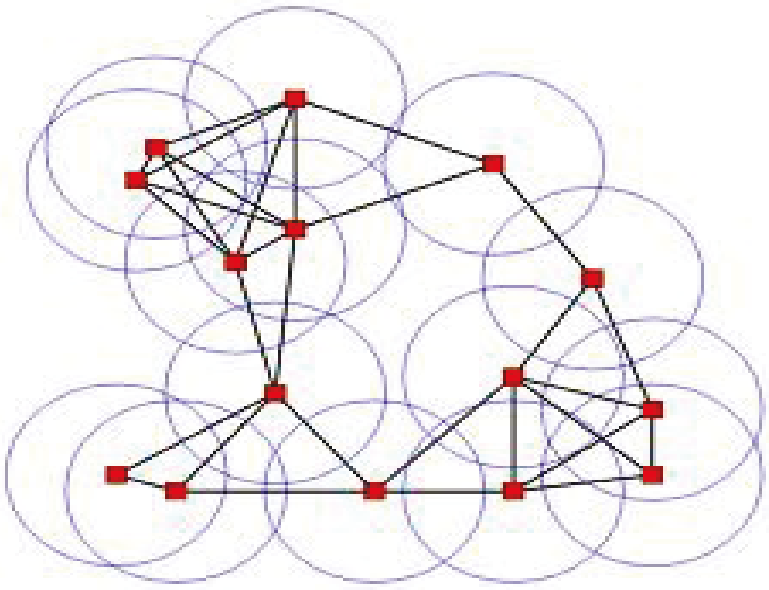

Figure 7.3: Unit disk graph

- the same low budget,
- different high budget: more budget for the looser,
- different low budget: more budget for the looser.

Table 7.2: Geometric graph :strategy U against strategy D

Probability to dominate opinions	strategy U	strategy D
equal and high budget (200/200)	72.9%	21%
equal and low budget(50/50)	49 %	47 %
strategy U has less and both with high budget (199/200)	66 %	26 %
strategy U has less and both with low budget (49/50)	45%	48 %

Strategy U against strategy D: From results in the table 7.2, by adopting the same high budget, strategy U can dominate more the network and obtain the majority of the positive reputation. However, when the forceful peer performing with strategy U but with less budget and low budget, the results are showing that the dominance will be for the strategy D with more budget. However, strategy U still dominates even with less budget for the high budget.

Strategy U against strategy $1/D$: From results in the table 7.3, by adopting the same high budget, strategy U can dominate more the network and obtain the majority of the reputations (more positive values) even with less budget for the high budget case. We show also that even by adopting small budget, the uniform strategy

Table 7.3: Geometric graph :strategy U against strategy 1/D

Probability to dominate opinions	strategy U	strategy 1/D
equal and high budget (200/200)	87%	9%
equal and low budget(50/50)	57.7 %	37 %
strategy U has less and high budget (199/200)	77 %	16 %
strategy U has less and low budget (49/50)	44 %	51.3 %

(strategy U) performs well against the inverse of degree strategy (strategy $1/D$). However, when the forceful peer performing with strategy U but with less budget for the low budget, the results show that the dominance will be for the strategy $1/D$ having more budget.

Table 7.4: Geometric graph: strategy U against strategy D^2

Probability to dominate opinions	strategy U	strategy D^2
high budget (200/200)	88 %	7%
low budget(50/50)	55.6 %	40 %
strategy U has less budget than strategy D^2 (199/200)	84.3 %	11.6 %
strategy U has less budget than strategy D^2(49/50)	52.7 %	42 %

Strategy U against strategy D^2: From results in the table 7.4, by adopting the same high budget, strategy U can dominate more the network. We show also that even by adopting small budget, the uniform strategy (strategy U) performs well against the strategy D^2. When the forceful peer is performing with strategy U but with less budget, the results are showing that the dominance will be also for the strategy U.

7.4.1.2 Strategies based on neighbors in geometric graphs

From results shown and explained previously, we affirm that uniform strategy, not taking into consideration the degree of the peer in the graph is dominating other strategies taking the degree as a factor. In the following, we want to compare strategies depending on the neighbors. For that we simulate three test-beds with different couples of strategies.

Table 7.5: Geometric graph : strategy D against strategy 1/D

Probability to dominate opinions	strategy D	strategy 1/D
equal and high budget (200/200)	40 %	50%
equal and low budget(50/50)	50.4 %	45.5%
strategy 1/D has less and high budget (200/199)	55.7 %	37.2%
strategy D has less and low budget (49/50)	37.2 %	57.2 %

Strategy D against strategy 1/D: When one peer adopts strategy D and an other the strategy 1/D, the table 7.5 shows that the first strategy outperforms the

strategy 1/D in the case of small budget but not for the high. When there is a difference between budgets, the strategy with high budget dominates the network of peers. This observation affirms the importance of the budget.

Table 7.6: Geometric graph: strategy D against strategy D^2

Probability to dominate opinions	strategy D	strategy D^2
equal and high budget (200/200)	48.6%	46%
equal and low budget(50/50)	81 %	13%
strategy D has less and high budget (199/200)	80 %	14 %
strategy D has less and low budget (49/50)	48 %	48.4 %

Strategy D against strategy D^2: When one peer adopts strategy D and an other the strategy D^2, the table 7.6 shows that the first strategy outperforms the strategy D^2 in the two cases of the small and the big budget.

Table 7.7: Geometric graph: strategy 1/D against strategy D^2

Probability to dominate opinions	strategy 1/D	strategy D^2
equal and high budget (200/200)	92 %	4%
equal and low budget(50/50)	52.7 %	42.9%
strategy 1/D has less and high budget (199/200)	91 %	4 %
strategy 1/D has less and low budget (49/50)	46.7 %	48.9 %

Strategy 1/D against strategy D^2: When one peer adopts strategy $1/D$ and an other the strategy D^2, the table 7.7 shows that the first strategy $1/D$ outperforms the strategy D^2 in the two cases: the small and the big budget. With a difference between budgets and using low one, strategy D^2 dominates but not for the high budget.

Table 7.8: Order of dominance for geometric graph

order	1	2	3	4
High budget	U	1/D	D	D^2
Low budget	D	U	1/D	D^2

As a conclusion about all strategies in geometric graph and through the results explained in tables, we affirm that:

- An order between strategies exists. It is shown in the table 7.8.
- For the low budget, strategy degree outperforms the strategy uniform that outperforms the strategy inverse of the degree that outperforms the square degree.
- For the high budget, strategy uniform outperforms the strategy inverse of degree that outperforms the strategy degree that outperforms the square degree.

- Strategy U dominates all the other strategies D, 1/D and D^2 that adopt the degree as a parameter to decide who affect.
- A comparison between strategies D, 1/D and D^2 reveals that strategy D performs well and better than strategy D^2 and then strategy$1/D$.
- For small budget, the nature of the networks (links and the transmission range) impacts the results. We conclude that when the budget in low and the strategies have very close performances, a decrease of budget can affect the results and inverse the results.
- The budget of affection is an important parameter, forceful peers with an important budget can dominate the network even with the not efficient strategy.
- The topology has more importance in the case of the low budget and the geometric graph. This observation justifies that degree outperforms the uniform strategy.

In the figure 7.4, we present an example of the output of a simulation of some peers under the pressure of two forceful peers. We obtain peers represented by "stars" and others with black points. Peers represented by stars are those dominated by the first forceful peer. The black points correspond to the peers dominated by the second forceful peer. In this example, the transmission range between peers is about 10.

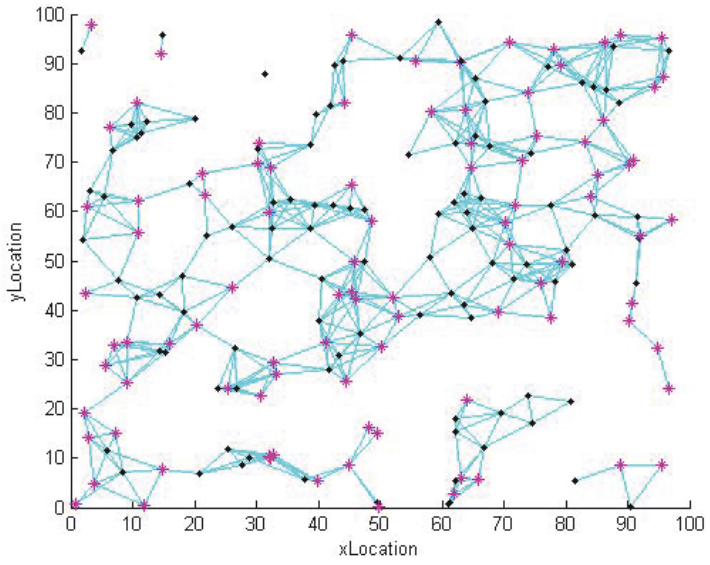

Figure 7.4: Example of results geometric graph

7.4.2 Random graph: ErdosRenyi graph

In the theory, the ErdosRenyi model is either of two closely related models for generating random graphs, including one that sets an edge between each pair of peers with equal probability, independently of the other edges. We generate a random graph based on the Erdos-Renyi model where all possible pairs of 100 nodes are connected with probability 0.1.

7.4.2.1 Uniform strategy in ErdosRenyi graph

In this section, an evaluation of the random uniform strategy is presented. To reach this purpose, we simulated matches between uniform strategy and the other strategies. suppose that one forceful peer is choosing the uniform strategy to impact the peers in this random graph, an other forceful peer is choosing strategy D or 1/D or D^2.

Table 7.9: ErdosRenyi graph :strategy U against strategy D

Probability to dominate opinions	strategy U	strategy D
equal and high budget (200/200)	46.7%	41.7%
equal and low budget(50/50)	48 %	44%
strategy U has less budget (high) (199/200)	44%	45%
strategy U has less budget (low) (49/50)	43%	49%

Strategy U against strategy D: From table 7.9, we observe that strategy U performs better than strategy D for the low and the high budget. When the budget of the forceful peer adopting strategy U is lower than the budget of the peer performing with strategy D, results are different: strategy D is dominating.

Table 7.10: Erdos-Renyi graph : strategy U against strategy 1/D

Probability to dominate opinions	strategy U	strategy 1/D
equal and high budget (200/200)	45%	42%
equal and low budget(50/50)	47.5%	46%
strategy U has less budget (high) (199/200)	44%	45%
strategy U has less budget (low) (49/50)	40%	53%

Strategy U against strategy 1/D: From table 7.10, we observe that strategy U performs better than strategy 1/D for the low and the high budget. When the budget of the forceful peer adopting strategy U is lower than the budget of the peer performing with strategy 1/D, results are different: strategy 1/D is the winner.

Strategy U against strategy D^2: From table 7.11, we observe that strategy U performs better than strategy D^2 for the low and the high budget. When we make the difference between the strategies in term of the budget, strategy D^2 is dominating for both the low budget but remains looser for the high budget.

Table 7.11: Erdos-Renyi graph : strategy U against strategy D^2

Probability to dominate opinions	strategy U	strategy D^2
equal and high budget (200/200)	50 %	38%
equal and low budget(50/50)	53 %	40.6%
strategy U has less budget (high) (199/200)	46%	42%
strategy U has less budget (low) (49/50)	45%	47 %

7.4.2.2 Strategy based on degree in ErdosRenyi graph

Table 7.12: Erdos-Renyi graph :strategy D against strategy 1/D

Probability to dominate opinions	strategy D	strategy 1/D
equal and high budget (200/200)	44.4%	44.9%
equal and low budget(50/50)	46%	44%
strategy 1/D has less budget (high) (200/199)	44%	42%
strategy D has less budget (low) (49/50)	41%	51%

Strategy D and strategy 1/D: From table 7.12, there is a difference between the low and the high budget: strategy D performs better than strategy 1/D for the low and almost equally for the high budget. When we make a difference between strategies in term of budget, strategy 1/D, having less budget, is dominating for the low budget case. For the high budget, D is dominating.

Table 7.13: ErdosRenyi graph :strategy D against strategy D^2

Probability to dominate opinions	strategy D	strategy D^2
equal and high budget (200/200)	45 %	42%
equal and low budget(50/50)	45 %	48%
strategy D has less budget (high) (199/200)	38%	47%
strategy D^2 has less budget (low) (50/49)	54%	37%

Strategy D against strategy D^2: From table 7.13, we show that strategy D^2 performs better than strategy D for the low but not for the high budget. When we make the difference between strategies in term of budget, strategy D^2, having more budget, is dominating for the high. In the case of low budget, the winner is strategy D having more budget than the strategy D^2 (who was the winner for equal budget)

Strategy 1/D and Strategy D^2 From table 7.14, we show that strategy 1/D performs better than strategy D^2 for the low but not with less budget and for the high budget even with less budget.

We conclude this part with some remarks and an example:

In this graph, we can resume the obtained results as following:

- An order between strategies exists. It is shown in the table 7.15.

Table 7.14: Erdos-Renyi graph : strategy 1/D against strategy D^2

Probability to dominate opinions	strategy 1/D	strategy D^2
equal and high budget (200/200)	47%	41%
equal and low budget(50/50)	48 %	44.7%
strategy 1/D has less budget (high) (199/200)	45%	41%
strategy 1/D has less budget (low) (49/50)	38%	53%

Table 7.15: Order of dominance for Erdos Renyi

order	1	2	3	4
High budget	U	D	1/D	D^2
Low budget	U	D^2	D	1/D

- Uniform strategy is the best strategy, it impact largely and randomly the peers in the network.
- Between strategy D, 1/D and D^2, results are very close and depend on the budget.
- For the high budget, the uniform strategy outperforms the degree and the inverse degree that outperform the square degree strategy.
- For the low budget, the uniform strategy outperforms the square degree strategy that outperforms the degree and the inverse degree.
- In this random graph, in some matches with different budget, the winner strategy for the equal strategy doesn't resist and looses. This due to the close strategies performances adopted by the two parties.

In the Erdos Renyi graph, the distribution of nodes degree follows the "poisson" one. So, the degree of nodes is almost around the average of the degrees. In this type of graphs, strategies that focus more on the nodes degree (strategy D^2) can dominate for the the low budget. However, when the budget is high, other strategies dominate strategy D^2. In fact, nodes have almost the same degree. So, concentrating on the same nodes in the case of the high budget can not be efficient.

In the Fig 7.5, we observe an example of the erdos-Renyi graph. Two colors are used to differentiate between peers dominated by the forceful peer (1) and the forceful peer (-1). We present here the random graph and the links between the peers.

7.4.3 Scale Free graph

A scale-free network is a network whose degree distribution follows a power law, at least asymptotically. Many networks are conjectured to be scale-free, including World Wide Web links, biological networks, and social networks, although the scientific community is still discussing these claims as more sophisticated data analysis techniques become available. We choose to generate a BarabásiAlbert (BA) [124]

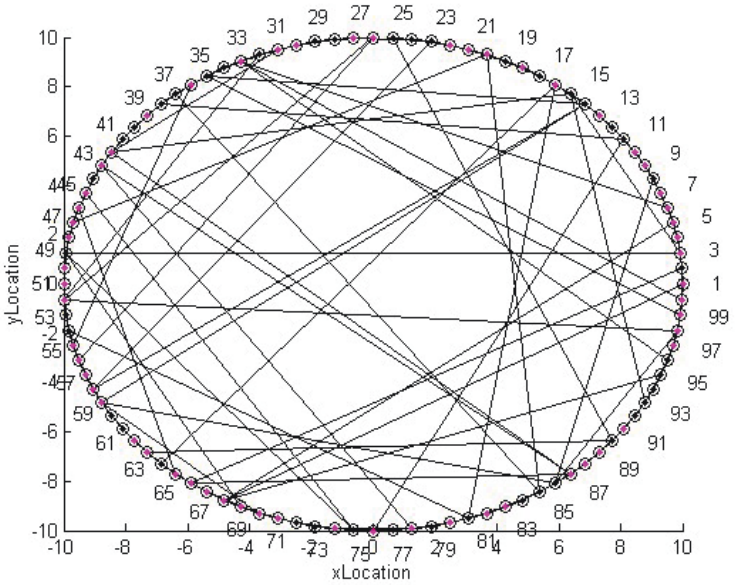

Figure 7.5: Example of results in erdos Renyi graph

scale free graph. BA model is an algorithm for generating random scale-free networks using a preferential attachment mechanism.

7.4.3.1 Uniform strategy in scale free graph

Table 7.16: scale free graph :strategy U against strategy D

Probability to dominate opinions	strategy U	strategy D
equal and high budget (200/200)	96%	2%
equal and low budget(50/50)	36 %	58 %
strategy U has less budget (high) (199/200)	93.3%	4%
strategy D has less budget (low) (50/49)	36.3%	57.8%

Strategy U against strategy D: In this table 7.16, we show that the strategy U (uniform distribution) against strategy D can obtain the majority of the peers in the network when adopting the big budget. However, in the case of the low budget, the uniform strategy looses against strategy D. To show the efficiency of the strategy, we decrease the budget for the winner in each case. We obtain the same dominance as before in the high and the low budget: uniform strategy dominates for the high budget and strategy D dominates for the low budget.

For this match between strategy U and strategy D, we observe that the dominance of strategy U is closely related to the budget. considering this information, simulations were done with different budget values. We obtain the Fig.7.6. With between 60 and 70 as a budget, strategy U begin to dominate the strategy D. So, it is the threshold budget related to this match between uniform and degree strategy.

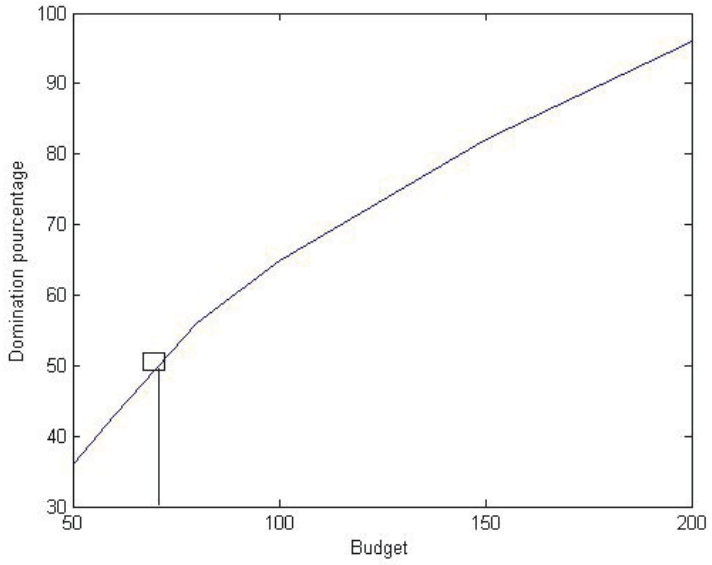

Figure 7.6: Threshold budget for strategy U against strategy D

Table 7.17: Scale free graph : strategy U against strategy 1/D

Probability to dominate opinions	strategy U	strategy 1/D
equal and high budget (200/200)	45.36%	45.22%
equal and low budget(50/50)	43%	50%
strategy 1/D has less budget (low) (50/49)	49.7%	44.6%

Strategy U against strategy 1/D: In this table 7.17, the strategy U and the strategy 1/D show almost the same results for the high budget 200. For the budget 50 (the low budget), a difference in the results can be observed. The strategy 1/D performs better than the strategy U. In case of strategy 1/D has less budget than the strategy U for the low, the strategy 1/D losses. The strategy 1/D is very related to the budget parameter in the scale free against the strategy U.

Table 7.18: Scale free graph : strategy U against strategy D^2

Probability to dominate opinions	strategy U	strategy D^2
equal and high budget (200/200)	100%	0%
equal and low budget(50/50)	94.5%	4 %
strategy U has less budget (high) (199/200)	100%	0%
strategy U has less budget (low) (49/50)	94%	0%

Strategy U and strategy D^2: In the table 7.18, results of simulations of the strategy U against the strategy D^2 in different cases are shown. In the first case of the high budget, the uniform method dominates with 100% even with less budget

than strategy D^2. For the second case, when the budget is low, results are almost the same: strategy U dominates with good percentage even with less budget than the strategy D^2.

After those three simulations, results are different and depends on the topology and the budget: for the high budget (200), uniform dominates the strategy of degree and the square of the degree but shows the same performance with strategy 1/D. For the low budget (50), the topology has more importance in the opinions management. For this reason, the strategy degree, D dominates the uniform, U.

7.4.3.2 Strategies based on degree in scale free graph

In this part, we simulate matches between strategy D, 1/D and D^2 in the scale free graph to obtain the order between them.

Table 7.19: Scale Free graph : strategy D against strategy 1/D

Probability to dominate opinions	strategy D	strategy 1/D
equal and high budget (200/200)	5%	90%
equal and low budget(50/50)	59.7 %	34%
strategy 1/D has less budget (high) (200/199)	7%	88.2%
strategy D has less budget (low) (49/50)	59.3%	34.7%

Strategy D against strategy 1/D: The table 7.19, we show the results when the strategy D is faced to strategy 1/D. In the case of high budget, strategy 1/D can obtain the majority of peers even with less budget. In the case of the low budget, strategy D performs well and better than strategy 1/D even with a less budget.

Table 7.20: Scale free graph : strategy D against strategy D^2

Probability to dominate opinions	strategy D	strategy D^2
equal and high budget (200/200)	100%	0%
equal and low budget(50/50)	95 %	3.5%
strategy D has less budget (high) (199/200)	100%	0%
strategy D has less budget (low) (49/50)	77.2%	19.5%

Strategy D against strategy D^2: The table 7.20, we show the results when strategy D is faced to strategy D^2. In all cases, strategy D is the winner. The nature of the graph is very important, it is more important than the strategy itself.

Strategy 1/D against strategy D^2: In the table 7.21, we show the results when the strategy 1/D is faced to strategy D^2. In all cases, strategy 1/D is the winner even with less budget than the strategy D^2.

As a conclusion in this graph, we can affirm that:

- An order between strategies exists. It is shown in the table 7.22.

Table 7.21: Scale Free graph : strategy 1/D against strategy D^2

Probability to dominate opinions	strategy 1/D	strategy D^2
equal and high budget (200/200)	100%	0%
equal and low budget(50/50)	91.7 %	6%
strategy 1/D has less budget (high) (199/200)	100%	0%
strategy 1/D has less budget (low) (49/50)	90 %	7%

Table 7.22: Order of dominance for Scale Free

order	1	2	3	4
High budget	U	1/D	D	D^2
Low budget	D	U	1/D	D^2

- For the low budget, the strategy degree outperforms the uniform and the inverse of the degree that outperform the quare degree strategy.
- For the high budget, the uniform and the inverse of the degree outperforms the degree strategy that outperform the quare degree strategy.
- The topology has more importance again in the case of the low budget. This observation justify that degree outperforms the uniform strategy. But, when the budget is high the network and the recommendations via neighborhood looses the impact.
- In this graph, strategies resist to the decrease of the budget in different matches: results show the same dominance when creating some little gap between strategy.

In the scale free graph, the distribution of node degree follows the power low. So, the probability to have node degree far from the average is high.. This explains the order between strategies seen in scale free graph. In fact, it is not efficient to put the maximum interests on nodes having high degree. For this reason, we observe that the strategy D^2 can't dominate against all other strategies.

In the figure 7.7, we present an example of the output of a simulation of some peers under the pressure of the two forceful peers. We obtain peers represented by "stars" and others with black points. Peers represented by stars are those peers dominated by forceful peer 1. The black points correspond to the peers dominated by forceful peer 2. Links between peers are represented by blue lines.

7.5 Conclusion

In this chapter, we designed a testbed to evaluate the impact of forceful peers through reputation model aggregation. This types of malicious peers can adopt some strategies to impact the reputation in a P2P network. We are showing that in different graphes, performances are different. Results affirm that an order between strategies exists and depends on the budget and the topology. When a forceful peer plays against an other

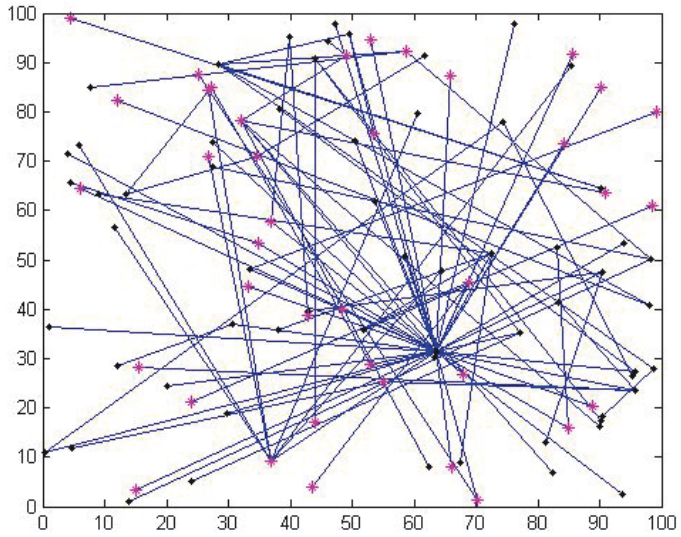

Figure 7.7: Example of results in scale free graph

forceful peer, it can happen that one of them know the strategy of the other and can build a suitable strategy. This will open new research aspects for our works.

Chapter 8

Conclusion and Ongoing Work

8.1 Conclusion

8.1.1 General conclusion

Trust and reputation management results from the incessant need to evaluate the participants in critical and open systems. It is important for many systems such as peer to peer sharing files, the cloud computing and distribute intrusion detection systems. However, this management has given birth to a large possibility to alter the normal functionalities of trust and reputation systems. That means that peers participating in reputation and trust management can adopt malicious behaviors: byzantine, rational, stubborn and forceful. This observation opens research and analysis to ameliorate the defense mechanism. In our contributions, we proposed a complete analysis of those malicious behaviors. We proposed also a defense mechanism to protect our reputation and trust system. The analysis and findings presented in this manuscript have shown a good performance for our reputation and trust system compared to known systems. We proposed also an optimized method to minimize the participation in the trust system for host intrusion detection. In addition, an aggregation method in distributed networks was proposed and validated. To achieve the analysis, we proposed a study of stubborn and forceful peers trying to spread misinformation into the proposed gossip-based reputation aggregation.

This chapter discusses the major conclusions of the work presented in the thesis.

8.1.2 Reputation system and its defense

In the first part, we proposed a trust-based reputation management system based on perron algorithm and credibility parameter.

Our system relies on interactions between peers to make the evaluations. Using evaluations and feedbacks, a central entity can estimate the trust of a given peer. Four approaches are proposed to estimate the trust of interacting peers: PerronTrust, CredTrust, CredTrust-trust and with pre-trusted peers. They are studied, simulated and compared between them and to two existing trust methods under several attack scenarios. Our analysis clearly shows that the third approach CredTrust-trust

combining the concepts of trust and credibility in a new way is the most efficient to detect malicious behaviors and to guide future executions in the hybrid cloud in term of selecting the dependable and reliable peers in cloud environment.

8.1.3 Refinement and use cases

On the basis of the good performance shown by the credTrust-Trust model in the first chapter, we have presented an improvement of our models: we investigate privilege idea. We have studied this refinement in the case of job allocation in cloud computing. We proposed also the benefit of our proposal to minimize the jobs loss in the same context. Then, we have introduced the peer to peer sharing files. We proved that with our reputation models, minimizing the inauthentic files is possible.

8.1.4 Optimization of collaboration

Afterward, we have investigated in the optimization of the resources for monitoring the network to evaluate each other. We have seen that to rate peers in the distribute network, peers must control the behaviors of other peers and this will consume resources. We have introduced for that problem the game theory model.

8.1.5 Aggregation and analysis of stubborn and forceful peers

Distributed nature of peer-to-peer networks and their large size make aggregation process complex. So, we have moved to a totally distributed network and we have proposed a gossip based aggregation model. Our mathematical framework stands out as a promising mathematical tool to evaluate the convergence in all cases. Analysis of stubborn peers, not respecting the update process, has also been done and numerical results have shown the validity of our results. Besides the stubborn peers, we have analyzed the effect of forceful peers with different strategies under different graphs (scale free, random graph and geometric graph). Results affirmed that an order between strategies exists and depends on the budget and the graph. This affirms the impact of this type of peers in the reputation system and opens new directions of research.

8.2 Future Directions

This thesis mainly concentrates on the design of a reputation management system, its defense mechanism, its applicability and the analysis of byzantine and forceful behaviors. Reputation management system for peer-to-peer networks raises many issues. In this thesis, we have selected few of them. Rest of the issues can be a part of further study. In specific, following issues can be further explored.

(1) While estimating trust, we have fixed parameters for example time taken to achieve the job. More parameters can be added by observing real system and consequently a better trust estimator can be obtained.

(2) We choose to show the benefit of our system in the cloud computing and the peer to peer network, we envision also to apply our secure model in social network.

(3) In gossiping it is assumed that nodes know the start of gossip. This is difficult to ensure. The proposed algorithm may also work for asynchronous case with minor modifications. These modifications and a theoretical proof for the algorithms can be pursued further.

(4) In the last part of the thesis, we are interested on applying the min cut model if one forceful peer has a knowledge about the other forceful peer strategy.

Appendices

Appendix A: Résumé

La confiance est un prérequis nécessaire à tout objet dans un système. Afin d'établir la confiance, des systèmes de gestion ont été développés pour pouvoir créer et faire propager la confiance au sein des communautés en ligne. Les systèmes de réputation sont des systèmes de gestion de confiance particuliers dont l'objectif est d'évaluer et d'attribuer des scores aux différentes ressources demandées et/ou utilisées dans une application. L'évaluation au sein d'un système de gestion de réputation dépend des participants. L'intention peut être purement informative comme elle peut être porteuse de menace d'attaque dans le but est de falsifier les scores. La force d'un système de réputation réside dans sa capacité à prévenir, détecter ces fraudes et les éliminer pour pouvoir générer le score le plus fiable possible.

A.1 Etat de l'art

L'objectif de cette partie est de présenter un état de l'art des principaux systèmes de réputation visant à les analyser et à montrer leurs forces et leurs limitations ainsi que les approches qu'ils adoptent à des fins d'évaluation.

L'objectif principal des modèles de confiance basés sur la réputation est d'identifier parmi les noeuds qui sont disponibles les plus fiables pour leur assigner des tâches ou leur envoyer des requêtes. Globalement, ces modèles suivent le même processus pour la sélection des noeuds de confiance. La première étape consiste à se baser sur sa propre expérience avec un nœud donné pour évaluer sa fiabilité ; c'est ce qu'on appelle la confiance directe. Cette confiance directe peut être évaluée en prenant en compte plusieurs facteurs comme : la qualité des précédentes transactions, leur nombre, l'importance de chacune d'entre elles et la satisfaction obtenue après chaque transaction. En plus de l'expérience personnelle, un pair peut utiliser les expériences des autres pairs (expérience indirecte) pour évaluer le degré de fiabilité d'un autre pair. Le crédit accordé aux expériences indirecte est différent de l'expérience personnelle d'un pair, des modèles comme Semantic Web [59] et Global Trust [31], font la différence entre la confiance faite à un pair en tant que fournisseur de service et en tant que recommandeur (expérience indirecte). Ils permettent ainsi de filtrer les pairs qui ont un comportement malveillant en notation (les pairs qui recommandent des pairs malveillants) des pairs qui ont un comportement malveillant en tant que fournisseurs de services (qui font des mauvaises transactions avec les autres pairs). La deuxième étape, est le calcul de la réputation en se basant sur les différentes expériences récoltées (directes et indirectes). A la fin de cette étape un pair décide

s'il effectue une transaction avec un autre pair ou pas. A la fin de chaque transaction, une réévaluation est faite en fonction du résultat de la transaction, et le fournisseur de service est évalué en fonction de la satisfaction du client (récompensé ou puni en conséquence). La réputation et la confiance sont évaluées de manière différente pour chaque modèle, certains utilisent les réseaux bayésiens (comme BNTBM [28]), d'autres utilisent la logique floue (fuzzy logic) (comme PATROL-F [30]), d'autres s'inspirent de la biologie (comme AntRep, TDTM [29] et TACS [27]), et d'autres donnent une expression analytique pour le calcul de la confiance (comme Global Trust et Semantic Web). Chacun de ses mécanismes a ses avantages et inconvénients, la logique floue permet de modéliser la réputation et la confiance de manière plus compréhensible à l'homme. Cependant elle est très difficile à implémenter sur de très grands réseaux (problème de passage à l'échelle). Les réseaux bayésiens, procurent une flexibilité qui permet de modéliser les différentes facettes de la confiance et les différents contextes de l'environnement. Mais cette technique est trés consommatrice en temps de calcul. Les expressions analytiques sont plus faciles à lire et à comprendre, mais ne fournissent pas beaucoup de possibilité pour la modélisation des différents facteurs impliqués dans la modélisation de la confiance. Enfin, les techniques inspirées de la biologie ont démontré leur eficacité lors du passage à l'échelle et leur adéquation dans les réseaux pair-à-pair. Cependant, dans certains cas leurs indéterminismes et leurs approximations conduisent à choisir des pairs malveillants comme étant les plus fiables. Dans ce qui suit, certains des systèmes cités ci-dessus seront présentés en détails.

A.1.1 CuboidTrust

CuboidTrust [62], est un modèle de confiance global pour les réseaux pair- à-pair basé sur la réputation, qui est construit autour de quatre relations basées sur trois facteurs de confiance : la contribution d'un pair au système (ressources), la fiabilité d'un pair (calculée à partir des retours des autres pairs : feedback) et la qualité des ressources mises à disposition par un pair. Chacun de ses facteurs est représenté par un vecteur formant un cuboide de coordonnées x; y; z et dénoté par Px;y;z , où z représente la ressource stockée par y et évaluée par x et Px;y;z l'évaluation de la ressource z par le pair x. Les valeurs attribuées à la qualité de la ressources sont binaires : +1 (positif) ou -1 (négatif). La ressource est évaluée positivement si à la fin d'un téléchargement sans incident la ressource est jugée intègre et authentique et évaluée négativement si elle n'est pas intègre.

A.1.2 EigenTrust

EigenTrust [113], est l'un des systèmes de réputation les plus cités et le plus utilisé comme référence dans les modèles de réputation des réseaux pair-à-pair. L'idée de EigenTrust est d'attribuer, à chaque pair, une note de confiance globale qui est calculée à partir des notes locales attribuées par chaque pair et pondérées par le taux de confiance en ces pairs. L'attribution des notes se base sur l'historique d'échanges entre les différents pairs. Le modèle EigenTrust est complétement distribué et permet d'assurer l'anonymat aux pairs qui attribuent les notes. Chaque transaction est évaluée localement par le pair i qui reçoit la ressource du pair j, notée $tr_{i;j}$ elle prend la valeur (+1) si la transaction est satisfaisante et (-1) sinon.

A.2 Le modèle de réputation proposé

Le calcul distribué et hybride est un procédé efficace pour de nombreuses applications. Cependant, il est difficile de gérer les risques impliqués dans l'interaction et la collaboration avec des pairs inconnus et potentiellement malicieux. Des modèles de gestion de la confiance et la réputation ainsi que leurs mécanismes de défenses sont décrits dans cette partie. Tout d'abord, en utilisant les résultats des évaluations distribués dans les pairs, une entité centrale collecte et évalue la fiabilité et la méchanceté de chaque participant. PerronTrust, CredTrust et CredTrust-trust sont les modèles de confiance dynamiques proposées. Pour assurer la lutte contre les comportements des noeuds malicieux, nous mettons en place la crédibilité des noeuds pour le calcul de la confiance dans les modéles CredTrust et CredTrust-trust. Un modèle analytique et la simulation de la performance sont proposés également. Nous analysons la résistance à d'éventuelles attaques comme les pairs collusoires. Nos modèles proposés fonctionnent bien et, en particulier, le modèle CredTrust-Trust combinant à la fois la crédibilité et la confiance.

A.2.1 L'architecture et les modèles d'attaques

A.2.1.1 L'architecture détaillée

L'architecture du système de réputation qu'on propose est illustrée dans la figure 1 qui est la figure de l'achitecture dans le chapitre 3. Les données qui sont utilisés pour établir la confiance sont enregistrés dans le "portal" qui calcule la confiance pour les noeuds et prend les décisions. Pour effectuer cette tâche primordiale, il possède trois modules: le collecteur des réputations, le gestionnaire de la confiance et le décideur.

Dans notre système, chaque noeud possède trois entités (dédiées à l'évaluation, à l'exécution des tâches et à l'envoi) puis un composant de stockage local.

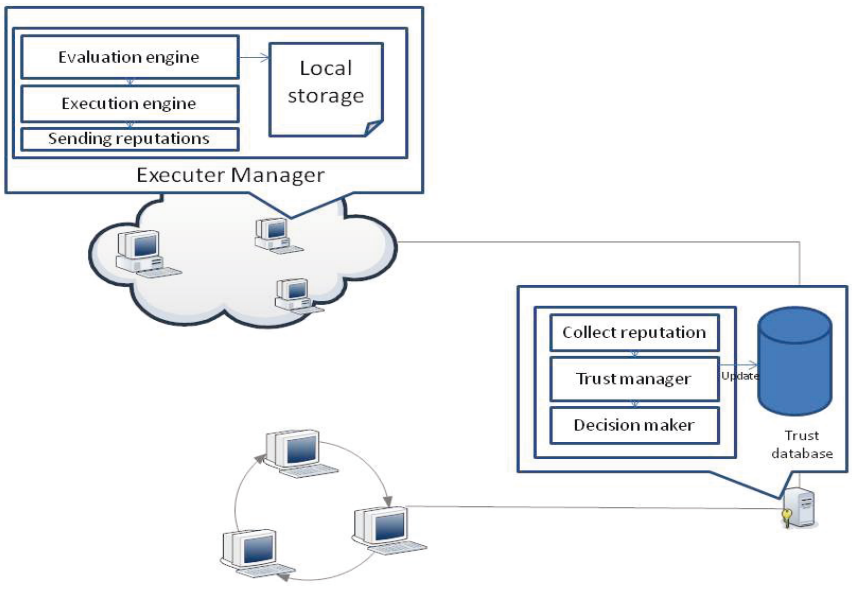

Figure 1: l'architecture détaillée

Table 1: Les paramétres du modéle

T	la matrice des réputations
T ij(t)	l'évaluation du noeud i pour j
Dij(t)	l'évaluation directe entre noeuds i et j
Delta t	la période de mise à jour
π	le vecteur de confiance
ϵ	limite de convergence
Threshold	le paramètre de décision
Cred	le vecteur de crédibilités

A.2.1.2 Les modèles d'attaques

Les noeuds exécuteurs dans notre système peuvent adopter deux comportements: normal ou tricheur. En se basant sur le modèle de BAR (Byzantine Altruiste Rational) [71], on classifie les attaques possibles en deux catégories : le comportement normal, les noeuds peuvent être altruistes, ce sont des noeuds qui suivent le protocole utilisé de manière exacte et le comportement tricheur, le noeud est soit :

• Rationnel : le nœud n'est intéressé que par l'optimisation de l'utilisation de ses ressources. Il ne déviera du protocole utilisé que si celui-ci estime qu'à cause du protocole utilisé ses performances diminuent;

• Byzantine : le nœud ne suit pas le protocole utilisé, soit parce qu'il est compromis, soit parce qu'il est mal configuré ou parce qu'il suit une fonction d'optimisation de ses ressources qui n'est pas la même qu'un noeud rationnel.

A.2.2 Résumé des modèles de réputation proposés

La table 1 résume les définitions des paramètres utilisés dans les algorithmes proposés. Notre première proposition est le modèle "PerronTrust" qui est basé sur la méthode de puissance et dans lequel la convergence vers un état stable est assurée par le théorème de Perron Frobenius. L'équation principale de ce modèle est la suivante :

$$\pi \leftarrow \frac{T^t.\pi}{||\pi||_1}. \tag{1}$$

Pour avoir une meilleure estimation de la réputation, on combine à chaque itération l'évaluation locale et la confiance des noeuds. Le calcul sera répété itérativement jusqu'à ce que le vecteur de la confiance devienne stable. Comme amélioration de ce modèle, on propose d'intégrer le paramètre "crédibilité". Dans notre système d'évaluation, le fait que des noeuds ont une bonne note de la confiance n'inclut pas qu'ils ont la capacité de bien évaluer les autres. Donc, il est important de donner moins d'importance aux évaluations de ces noeuds.

La valeur de la crédibilité va décroitre si le noeud donne des fausses évaluations concernant les autres. Avec cette valeur réduite, l'impact de ces évaluations sera moins important. On peut évaluer la crédibilité globale selon la formule suivante (citée dans le chapitre 3):

$$\mathrm{Cred}_i = 1 - \left(\frac{\sum_{j=1}^{N} |\pi_j - T_{ij}|^2}{\sum_{j=1}^{N} |\pi_j - [1-\pi_j]|^2} \right)^{\alpha} \quad (2)$$

où $[1-\pi_j]$ désigne le nombre entier le plus proche $(1-\pi_j)$ et α est le nombre fixé. Notons que $[1-\pi_j]$ est égal à 0 si $\pi_j > 0,5$ et 1 si $\pi_j < 0,5$. Pour estimer le vecteur de la confiance, on utilise la formule suivante (citée dans le chapitre 3):

$$\pi \leftarrow \frac{T^t.Cred}{||Cred||_1} \quad (3)$$

Comme les noeuds malveillants donnent les bonnes évaluations dans presque tous les cas, ils auront une grande crédibilité. Cela signifie que l'évaluation donnée par ces pairs malveillants aura plus d'impact sur le résultat final. Pour surmonter ce problème, nous allons réintroduire à nouveau la confiance dans le processus itératif. Au lieu d'utiliser seulement la crédibilité (comme dans CredTrust) ou la confiance (comme dans PerronTrust), nous combinons les deux.

Plus précisément, nous utilisons la formule suivante (citée dans le chapitre 3):

$$\pi \leftarrow \frac{T^t(Cred \otimes \pi)}{||Cred \otimes \pi||_1} \quad (4)$$

où $Cred \otimes \pi$ désigne le produit composante par composante de $Cred$ et π. Il est maintenant clair que, même si un noeud a une grande crédibilité, l'impact de son opinion est atténué comme il a une confiance faible.

A.2.3 Résumé des résultats obtenus

Prenons par exemple les résultats de la simulation des noeuds qui trichent en inversant les évaluations sur les autres noeuds. Supposant que ces noeuds tricheurs évaluent les autres noeuds tricheurs avec 0.8 et les non-tricheurs avec 0.2. Les noeuds normaux évaluent les noeud tricheurs avec 0.2 et les non-tricheurs avec 0.8.

La figure 2, du chapitre 3, montre la confiance estimée pour les noeuds malicieux (tricheurs). Quand le pourcentage des noeuds tricheurs est d'environ 10%, les quatre algorithmes donnent les mêmes estimations. Mais, quand ce pourcentage augmente, l'algorithme CredTrust-trust donne une meilleure estimation des noeuds tricheurs. PerronTrust et AverageTrust ne punissent pas les noeuds qui donnent de fausses évaluations car il ne considérent pas le paramètre de crédibilité. Par contre, CredTrust-trust et CredTrust montrent une meilleure performance par rapport aux autres grâce à la crédibilité qui pénalise les tricheurs.

D'autres simulations avec différentes attaques sur le système de confiance et des comparaisons avec deux modèles existants bien connus (EigenTrust et averagetrust) ont été effectuées. Les résultats de la première partie d'expériences montrent que le modèle proposé sélectionne les pairs fiables dans les nuages.

La deuxième partie de simulations a également confirmé que la crédibilité est bien adaptée pour mettre en évidence les comportements des pairs malveillants (Byzantins) et rationnels.

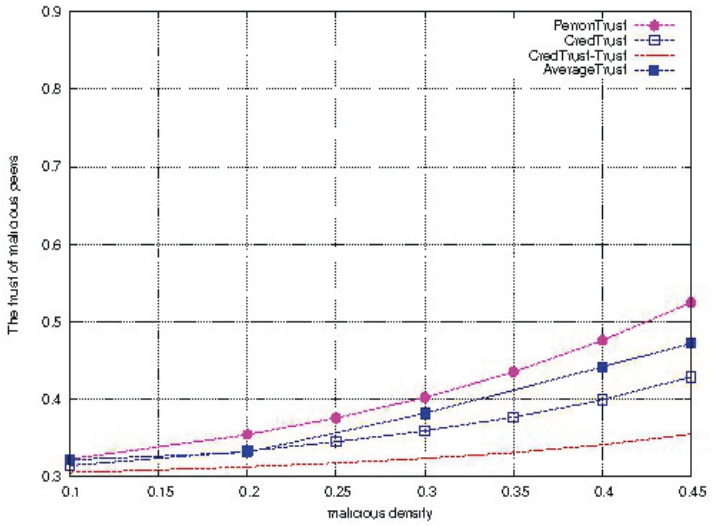

Figure 2: Les résultats de la confiance pour les noeuds byzantins

Nous avons également remarqué que de nombreuses attaques collectives peuvent être clairement détectés et évités avec une grande précision. Cela est particulièrement vrai quand CredTrust-trust est utilisé. Ce dernier a des performances nettement meilleures que EigenTrust et AverageTrust. La combinaison de la confiance et de la crédibilité améliore la performance du modèle d'une manière très significative. En outre, la complexité de l'algorithme est très limitée, ce qui lui permet d'être utilisé dans des systèmes à grande échelle.

A.3 Validation dans le cas d'allocation de ressources et de tâches dans le cloud et le partage de fichier dans les réseaux

Nous proposons dans cette contribution une méthode d'allocation de ressources dans les nuages et un partage de fichiers en réseau P2P basés sur la confiance. Cette contribution est divisée en deux parties : Dans la première partie, l'allocation de ressources utilise les réputations des utilisateurs. Ceci est important lorsque les ressources sont réparties dans le réseau comme dans notre cas. Dans la deuxième partie, nous utilisons le simulateur de cycle de requêtes et de partage de fichiers P2P. Nous avons modifié le simulateur pour implémenter notre modèle. Dans cette modification, l'algorithme CredTrust-Trust est utilisé pour attribuer des valeurs de confiance sur la réussite d'un téléchargement de fichier. L'objet de cette contribution est de montrer l'utilité et l'efficacité de notre modèle.

L'avantage de l'utilisation du modèle proposé dans le partage de fichier

Dans les simulations, pour prouver les bienfaits de la confiance, supposons que les noeuds malicieux partagent toujours des fichiers non authentiques quand ils sont sélectionnés comme source. Dans notre simulation, nous avons considéré un réseau de 53 noeuds honnêtes dont 5 sont initialement jugés comme digne de confiance. On obtient les valeurs de confiance représentées dans la figure 3 prise du chapitre 4 de ce manuscrit. En se basant sur ces valeurs, les sources de téléchargement sont sélectionnées dans le simulateur.

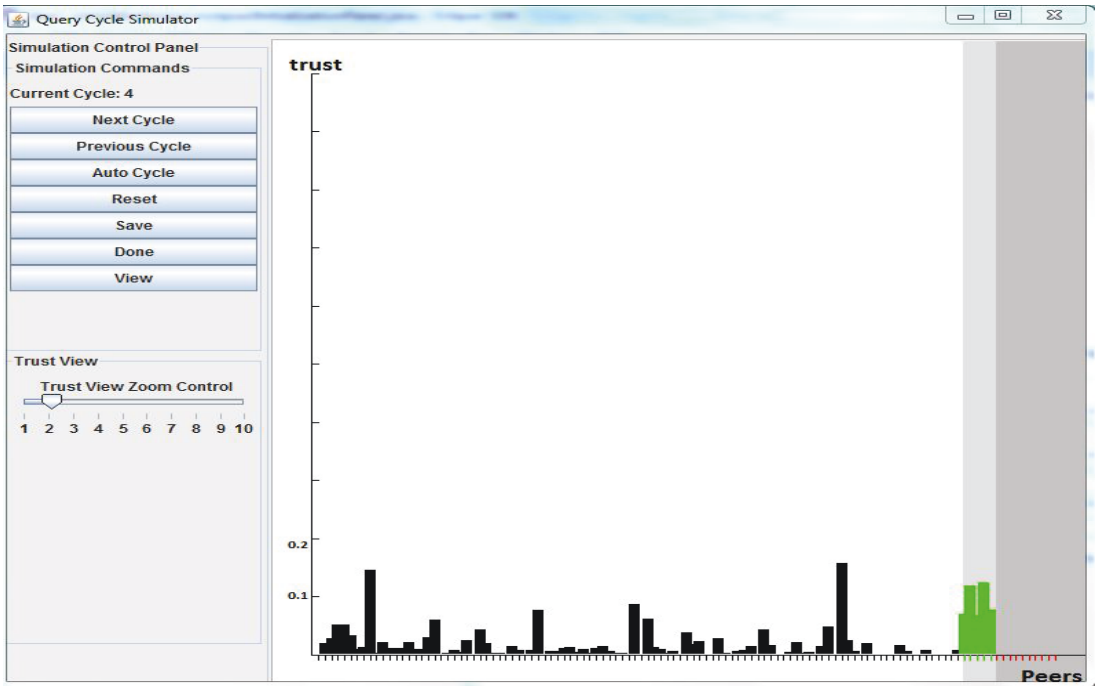

Figure 3: Résultats des valeurs de la confiance

Dans chaque expérimentation, on ajoute au réseau un nombre des noeuds malicieux qui doit rester inférieur à 40% de la somme totale des noeuds.

Les résultats sont illustrés dans la figure 4 du chapitre 4.

A.4 Une approche basée sur la théorie de jeux

Nous avons conçu une nouvelle plateforme basée sur la théorie des jeux et l'élection pour améliorer la détection des noeuds malveillants, optimiser les ressources et faire face à des domaines hétérogènes. Ici, le premier acteur est l'IDS et le second est le noeud malveillant. Les IDS tentent d'améliorer la détection alors que les noeuds malveillantes essayent d'augmenter les attaques. Nous montrons que la confiance et la réputation constituent de grands défis lorsque nous avons différents participants. Notre plateforme basée sur la théorie des jeux, avec la gestion de la confiance et de l'élection efficace, peut réduire la consommation des ressources et améliorer la gestion de l'évaluation.

Dans la figure 5, qui correspond à la figure du flus dans le chapitre 5, nous montrons la plateforme proposée. Il y a trois étapes:

- Etablissement de la réputation entre les HIDS
- Election de l'HIDS principal
- Détection en se basant sur la réputation et la théorie des jeux

A.5 Le modèle "gossip" pour la gestion de réputation

Dans notre première contribution, nous supposons que nous avons une entité centrale qui gère l'agrégation. Cette entité centrale regroupe les réputations et estime la confiance. Mais, nous voulons adopter un système totalement distribué pour permettre à chaque noeud d'estimer la confiance de l'autre. Toutefois, les

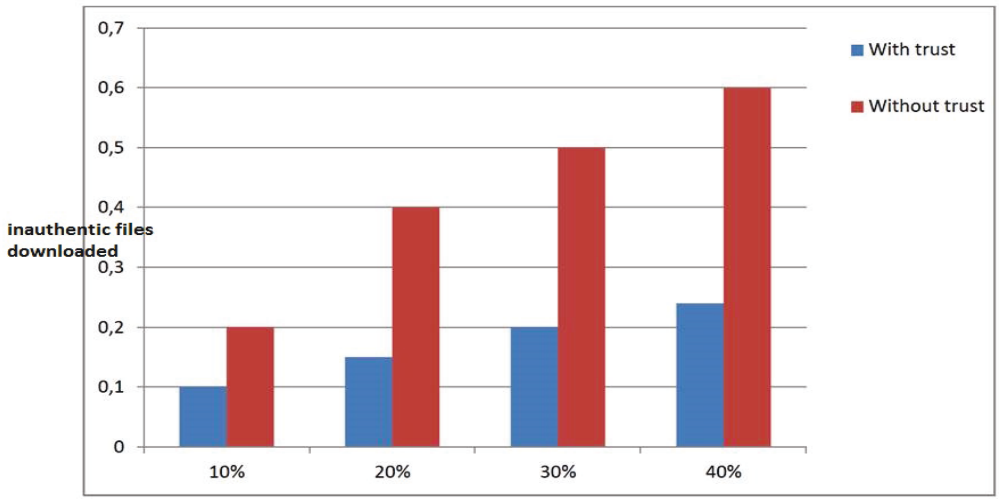

Figure 4: les téléchargements inauthentiques

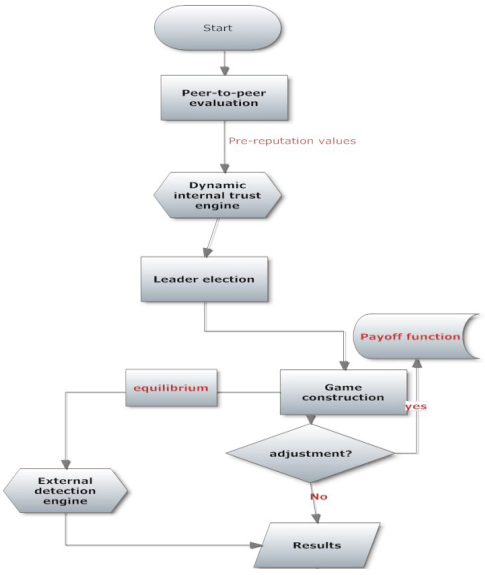

Figure 5: le diagramme de flux

Table 2: Définitions

Notation	definition
d_{vw}	l'évaluation locale de v pour w
R^v	la réputation globale de v
numOfPeers	nombre de noeuds
deg(v)	nombre de voisins de v

noeuds malveillants peuvent apparaître comme têtus et stratégiques. Donc, de bonnes méthodes d'agrégation de confiance doivent être conçus pour un système de gestion de réputation efficace. Dans cette contribution, nous décrivons la gestion de la réputation dans le cas normal (tous les noeuds suivent le modèle) et le cas de noeuds têtus . Nous mettons en oeuvre et validons notre proposition mathématique dans le cadre d'un exemple. Nous analysons la propagation de la désinformation et la persistante du désaccord en cas de comportements têtus. Le paramétres à utiliser dans modèle proposé sont résumés dans la tableau 2.

Algorithm 20: Mise à jour de R^v

1: If $w \sim v$ then
2: $R_{wv} \leftarrow \alpha.d_{wv} + \frac{(1-\alpha)}{deg(w)-1} \sum_{k\sim w, k\neq v} R_{kv}$
3: If $w \nsim v$ then
4: $R_{wv} \leftarrow \frac{1}{deg(w)} \sum_{k\sim w, k\neq v} R_{kv}$

Dans cette partie, on analyse le modèle dans le cas où les noeuds sont coopératifs et participent à la mise à jour de la réputation. Fixons un noeud et observons la mise à jour de son vecteur de réputation global. On peut écrire la colonne qui correspond au noeud v comme ce qui suit: $R^v = A^v R^v + h$, tel que $h = \alpha * d$

Theorem 5 *Le vecteur de réputation global, R^v_∞, converge vers $(I - A^v)^{-1} * h$, où I est la matrice d'identité.*

Maintenant, on s'intéresse au cas ou il y a des noeuds têtus. Pour ce cas, le réseau sera composé de l'union des noeuds coopératifs et des têtus. On peut représenter ce réseau comme suit:

$$N = \begin{bmatrix} C \\ NC \end{bmatrix}$$

Le vecteur de réputation lui aussi sera représenté différamment comme suit:

$$R^v = \begin{bmatrix} R^v_C \\ R^v_{NC} \end{bmatrix}$$

On peut affirmer aussi que R^v_{NC} sera une constante. Les noeuds têtus ne participent pas à la mise à jour des évaluations. Pour la matrice A^v, elle sera représentée comme suit:

$$N = \begin{bmatrix} A^v_C & A^v_{NC} \\ 0 & I \end{bmatrix}$$

Pour la convergence dans le cas des noeuds têtus, on a le théorème suivant:

Theorem 6 *$R^{v,n+1} = A^{v,n}R^{v,n}$ converge vers $R^v_\infty = (I - A^v_C)^{-1}(h + A^v_{NC} * R^v_{NC})$*

Aussi, un petit aperçu du réseau dans lequel on a implémenté et validé le modèle:

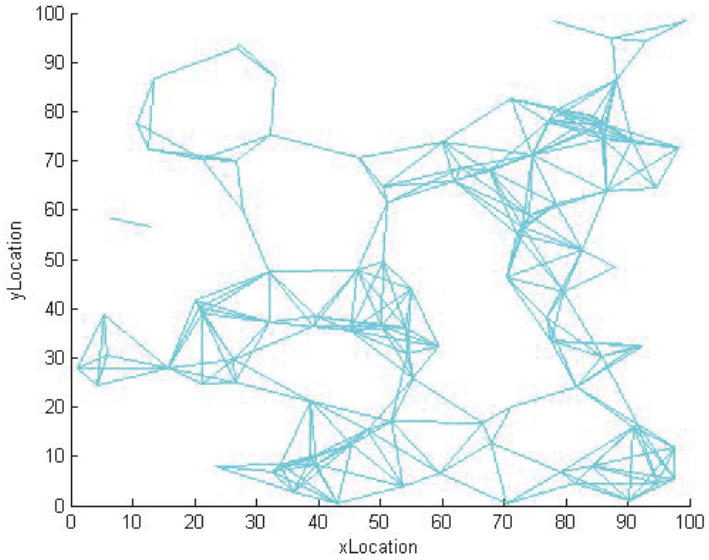

Figure 6: Un exemple de 100 noeuds

Dans cet exemple réseau pris du chapitre 6, on a mis en place la mise à jour du vecteur réputation jusqu'à la convergence selon l'algorithme proposé 21 du chapitre 6. Puis on implémente et on calcule R^v_∞ et on compare.

Algorithm 21: La réputation glabale

Require: numOfPeers
Ensure: R^v
1: Initialize the global reputation matrix, R
2: **repeat**
3: $T \leftarrow R^v$
4: **for** $v \in 1..numOfPeers$ **do**
5: update (R^v)
6: **end for**
7: **until** $T - R^v < \epsilon$

A.6 Etude de comportements des noeuds stratégiques dans différents graphes

Les noeuds stratégiques peuvent influencer la gestion de la réputation dans les réseaux distribués. On peut interpréter ces noeuds comme des leaders qui ont un effet sur les opinions. Il peuvent être la force du média qui agit sur le public. Ces noeuds adaptent des stratégies pour manipuler les évaluations. On propose quatre stratégies. Les notations utilisés sont résumées dans le tableau 3.

Table 3: Définitions

Notation	définition
d_i	la réputation locale du noeud i
R	la réputation globale
BMAX	le budget maximal
Re	remaining resource after each affectation
deg(i)	the degree of the peer i
numOfPeers	le nombre des noeuds
Aff_-	les affectations du noeud (-1)
$nAff_+$	le nombre d'affectations du noeud (1)
Aff_+	les affectations du noeud (1)
$nAff_-$	le nombre d'affctations du noeud (-1)

A.6.1 Les stratégies

Pour la description des stratégies, on commence par la stratégie uniforme (Alg.22 pris du chapitre 7). Ce mécanisme consiste à choisir un nombre aléatoire qui va être le numéro du noeud à affecter. Il lui attribue le poids et diminue ainsi son budget. Cette attribution se répète jusqu'à l'épuisement de la totalité du budget.

Algorithm 22: Strategy U: Random uniform

Ensure: Affectation for each peer, $Aff_* : Aff_- or Aff_+$
Require: numOfPeers
1: **while** $R < BMax$ **do**
2: Generate a random number x between 1 and numOfPeers
3: $Aff_*(x) = Aff_*(x) + 1$
4: $nAff_* = +1$
5: Re=Re-1
6: **end while**

Quand l'affectation des noeuds dépend des voisinages (les degrés des noeuds), on parle de la deuxième stratégie (Alg.23 cité dans le chapitre 7). Dans cet algorithme, on calcule les probabilités $P_i = \frac{deg(i)}{sum(deg)}$ et les intervalles correspondants $[P_i, P_i + P_{i+1}]$ pour tous les noeuds i. Comme l'autre stratégie, il y a un tirage aléatoire, affectation selon les degrés et une répétition jusqu'à la fin du budget.

Dans l'algorithme Alg.24 (cité dans le chapitre 7) qui correspond à la troisième stratégie, on calcule $P_i = \frac{1/deg(i)+1}{sum(1/deg+1)}$ pour privilégier les noeuds qui ont moins de degrés.

Dans l'algorithme Alg.25(cité dans le chapitre 7) qui correspond à la dernière stratégie proposé, on calcule $P_i = \frac{deg(i)^2}{sum(deg^2)}$ pour privilégier de plus en plus les noeuds qui ont plus de voisins.

Après avoir présenter les strategies que les noeuds (1) et (-1) vont adapter, on présente les résultats classés par type de graphe.

A.6.2 Les graphes géométriques

- Un ordre entre les stratégies existe 4.

Algorithm 23: Strategy D: Degree

Ensure: Affectation for each peer, $Aff_* : Aff\_or Aff_+$
Require: numOfPeers
1: **for** $i \in 1..numOfPeers$ **do**
2: calculate $P_i = \frac{deg(i)}{sum(deg)}$
3: **end for**
4: Determine the intervals $[P_i, P_i + P_{i+1}]$
5: **while** $R < BMax$ **do**
6: Generate a random number x between 0 and 1
7: Determine i verifying $x \in [P_i, P_i + P_{i+1}]$
8: $Aff_*(i) = Aff_*(i) + 1$
9: $nAff_* = +1$
10: Re=Re-1
11: **end while**

Algorithm 24: Strategy 1/D : 1/Degree

Ensure: Affectation for each peer, $Aff_* : Aff\_or Aff_+$
Require: numOfPeers
1: **for** $i \in 1..numOfPeers$ **do**
2: calculate $P_i = \frac{1/deg(i)+1}{sum(1/deg+1)}$
3: **end for**
4: Determine the intervals $[P_i, P_i + P_{i+1}]$
5: **while** $R < BMax$ **do**
6: Generate a random number x between 0 and 1
7: Determine i verifying $x \in [P_i, P_i + P_{i+1}]$
8: $Aff_*(i) = Aff_*(i) + 1$
9: $nAff_* = +1$
10: Re=Re-1
11: **end while**

Algorithm 25: Strategy D^2: $Degree^2$

Ensure: Affectation for each peer, $Aff_* : Aff\_or Aff_+$
Require: numOfPeers
1: **for** $i \in 1..numOfPeers$ **do**
2: calculate $P_i = \frac{deg(i)^2}{sum(deg^2)}$
3: **end for**
4: Determine the intervals $[P_i, P_i + P_{i+1}]$
5: **while** $R < BMax$ **do**
6: Generate a random number x between 0 and 1
7: Determine i verifying $x \in [P_i, P_i + P_{i+1}]$
8: $Aff_*(i) = Aff_*(i) + 1$
9: $nAff_* = +1$
10: Re=Re-1
11: **end while**

Table 4: Order of dominance for geometric graph

order	1	2	3	4
High budget	U	1/D	D	D^2
Low budget	D	U	1/D	D^2

Table 5: Order of dominance for Erdos Renyi

order	1	2	3	4
High budget	U	D	1/D	D^2
Low budget	U	D^2	D	1/D

- Stratégie U domine toutes les autres stratégies D, $1/D$ et D^2 qui adoptent le degré des noeuds comme un paramètre pour décider l'affectation.
- La comparaison entre les stratégies D, 1 / D et D^2 révèle que la stratégie D fonctionne bien et mieux que la stratégie D^2, puis la stratégie $1/D$.
- Pour un petit budget, la nature des réseaux (liens et les transmissions) a une incidence sur les résultats.

A.6.3 Les graphes aléatoires "Erdos Renyi"

Un exemple de graphe est donné dans la figure 7 citée dans le chapitre 7. Dans ce graphe, les résultats obtenus sont les suivants:

- un ordre entre les stratégies existe. Il est représenté dans le tableau 5
- la stratégie uniforme est la meilleure stratégie qui impacte largement les noeuds dans le réseau.
- entre les stratégies D, 1/D et D^2, les résultats sont très proches et dépendant du budget.
- pour le budget élevé, la stratégie uniforme surpasse la stratégie degré et celle du degré inverse qui surpasse celle du degré carré.
- pour le budget le moins élevé, la stratégie uniforme surpasse celle du degré carré qui surpasse celle du degré et celle du degré inverse.

A.6.4 Les graphes à grande échelle "Scale Free"

Les résultats obtenus pour ce graphe sont les suivants:

Table 6: Order of dominance for Scale Free

order	1	2	3	4
High budget	U	1/D	D	D^2
Low budget	D	U	1/D	D^2

- L'ordre obtenu est décrit dans le tableau 6.

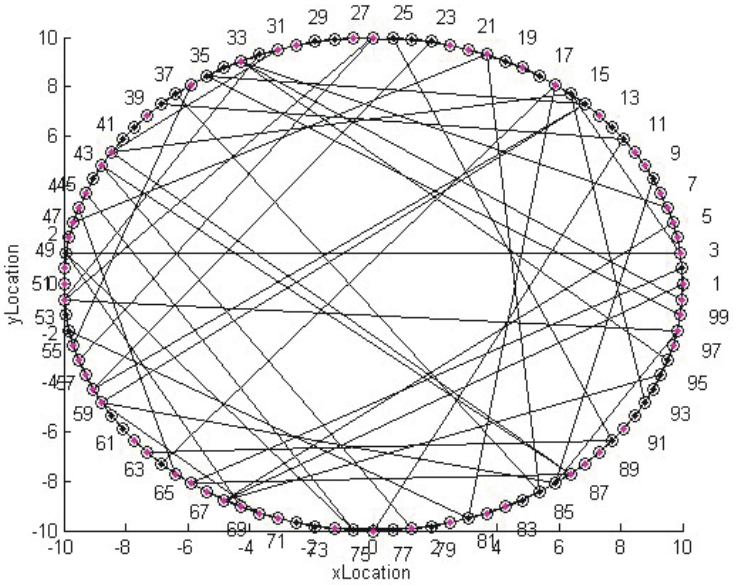

Figure 7: Exemple de résultat dans erdos Renyi

- pour le budget le moins élevé, la stratégie degree surpasse celle du uniforme et celle de degré qui surpassent celle du degré carré.
- pour le budget élevé, uniforme et celle de degré qui surpassent celle du degree puis celle de degré carré.

A.6.5 Analyses

Les résultats montrent qu' un ordre existe entre les stratégies et dépendent du budget et du type de graphes. Dans les graphes aléatoires de type Erdos-Renyi, la distribution de degré de noeud suit la loi de poisson. Donc, les degrés des noeuds sont pratiquement autour de la moyenne. Dans ce type de graphes, les stratégies qui se concentrent davantage sur le degré de noeud (stratégie D^2) peuvent dominer dans le cas du faible budget. Mais, lorsque le budget est élevé, d'autres stratégies dominent la stratégie D^2. C'est parce que les noeuds ont presque les mêmes degrés. Alors une concentration sur le même noeud dans le cas de gros budget ne peut pas être efficace.

Dans les graphes "scale free", la distribution des degrés des noeuds suit la loi de puissances. Ainsi, la probabilité d'avoir un degré de noeud loin de la moyenne est élevée. Le nombre de noeuds ayant un haut degré est très faible. C'est ce qui explique l'ordre entre les stratégies observées dans le graphe. En fait, il n'est pas efficace de mettre le maximum d'intérêt sur des noeuds ayant un degré élevé. Pour cette raison, nous observons que la stratégie D^2 ne peut pas dominer contre toutes les autres stratégies.

A.7 Conclusion

La gestion de la confiance et de la réputation vient du besoin d'évaluer les participants dans le système ouvert et distribué. Mais, la gestion coopérative ouvre la possibilité des attaques qui visent la modification du déroulement normal du système.

Ces attaquants peuvent adopter plusieurs comportements comme celui du byzantin, rationnel, têtu, stratégique. Pour lutter contre ces comportements, il faut les étudier et proposer des solutions de défense.

Dans cette thèse, on a proposé des analyses complètes de tous les comportements possibles dans le cas de deux systèmes proposées l'un centralisé et l'autre complétement distribué. Des solutions de défense ont été proposés pour punir les comportements malveillants des noeuds participants à l'évaluation. Les simulations et les résultats ont montré de bonnes performances obtenues grâce à notre modèle. On a proposé une gestion de réputation basée sur la confiance. La convergence de ce modèle est assuré par l'algorithme de perron. Pour assurer un meilleur système de défense, on a proposé le paramètre crédibilité pour calculer la réputation des noeuds. Nos algorithmes ont montré un bonne performance en les comparant à d'autres modéles et en les simulant avec plusieurs possibilités d'attaques.

Comme validation, on a montré l'utilité de notre modèle dans le cas d'allocation de tâches dans les nuages informatiques. On a également montré l'avantage de la réputation dans la diminution de la perte de tâches si on les laisse allouées à des noeuds malicieux. Dans le cas du partage des fichiers dans les réseaux distribués, notre modèle contribue à la diminution des fichiers non authentique au choix des noeuds malicieux comme source de téléchargement.

Après ces simulations, on a voulu optimiser la coopération pour évaluer les noeuds dans le système. En effet, l'opération d'évaluation consomme des ressources. On a proposé une plateforme à trois niveaux, évaluation, élection et contrôle basée sur la théorie de jeux. Le fait de décider si on doit évaluer ou pas grâce à la théorie de jeux va réduire la consommation des ressources des noeuds.

Pour la dernière partie, on a proposé un modèle d'agrégation totalement distribué pour la gestion de la réputation. Ce modèle a été étudié mathématiquement (convergence et démonstration) puis validé avec l'implémentation. On a étudié a travers des exemples et théoriquement le cas où il y a des noeuds têtus qui veulent falsifier les réputations. On a proposé aussi une étude complète des noeuds stratégiques avec des stratégies possibles et des simulations dans trois types de graphes. Ces noeuds peuvent influencer les opinions et cette influence dépend de la stratégie, des capacités et des réseaux.

Comme perspectives, on appliquera nos modèles dans les réseaux sociaux *où* la gestion d'opinion est très importante. Aussi, on pense à continuer l'analyse des noeuds stratégiques en supposant qu'il y a deux noeuds qui influent le réseau mais l'un connait la stratégie de l'autre. Dans ce cas, on proposera une nouvelle stratégie adapté par un noeud pour neutraliser l'effet de l'autre noeud stratégique.

Bibliography

[1] P.Zhang and al. "Survey of trust management on various networks", International Conference on Complex Intelligent and Software Intensive Systems, 2011.

[2] Marsh, S., "Formalising Trust as a Computational Concept. PhD thesis", University of Stirling, Department of Computer Science and Mathematics, 1994.

[3] A. Mejia et al., "A game theoretic trust model for on-line distributed evolution of cooperation in MANETs" Elsevier Journal of Network and Computer Applications.34:39-51, 01/2011.

[4] C.Dmitry R.Julian and W.Thomas "Into the Cloud, an evaluation of the google app engine", Report 2010

[5] S. Ganeriwal, L.K. Balzano, and M.B. Srivastava, "Reputation-based framework for high integrity sensor networks," ACM Transitions on Sensor Network, vol. 4, no. 3, May 2008.

[6] K. Liu, N. Abu-Ghazaleh, and K.-D. Kang, "Location verification and trust management for resilient geographic routing," J. Parallel and Distributed Computing, vol. 67, no. 2, pp. 215-28, 2007.

[7] R.A. Shaikh, H. Jameel, B.J. d'Auriol, H. Lee, S. Lee, and Y.J. Song, "Group-based trust management scheme for clustered wireless sensor networks," IEEE Transactions on Parallel and Distributed Systems, vol. 20, no. 11, pp. 1698-1712, Nov.2009.

[8] T.Kim ,and H.Seo, "A trust model using fuzzy logic in wireless sensor network", World academy of science, engineering and technology, 42, 2008.

[9] K. Hoffman, D. Zage, and C. Nita-Rotaru, "A survey of attack and defense techniques for reputation systems", technical report, Purdue Univ., 2007. http://www.cs.purdue.edu/homes/zagedj/docs/reputation survey.pdf

[10] M. Abadi, M. Burrows, B. Lampson, and G. Plotkin, "A calculus for access control in distributed systems," ACM Transactions on Programming Languages and Systems, vol. 15, no. 4, pp. 706 734, 1993

[11] G. Zacharia, A. Moukas, and P. Maes, "Collaborative reputation mechanisms for electronic marketplaces," in Proc. of the 32nd Annual Hawaii Int. Conf. on System Sciences, 1999.

[12] Z.Malik and A. Bouguettaya, "Reputation bootstrapping for trust establishment among web services," IEEE Internet Computing, vol. 13, no. 1, pp. 4047, 2009

[13] A. Joang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," Decision Support Systems, vol. 43, no. 2, pp. 618644, Mar 2007

[14] P. Laureti, L. Moret, Y.-C. Zhang, and Y.-K. Yu, "Information filtering via iterative refinement," in Europhysics Letters, vol. 75, no. 6, 2006, pp. 10061012.

[15] J. Zhang and R. Cohen, "A personalized approach to address unfair ratings in multiagent reputation systems," in Proc. of the Fifth Int. Joint Conf. on Autonomous Agents and Multiagent Systems (AAMAS) Workshop on Trust in Agent Societies, 2006.

[16] A. Josang and R. Ismail, "The beta reputation system," in Proc. of the 15th Bled Electronic Commerce Conf., 2002

[17] Y.Wang, J.Vassileva. Trust and Reputation Model in Peer-to-Peer Networks. In Proceedings of International Conference on Peer-to-Peer Computing (P2P'03), IEEE, 2003.

[18] S. Kamvar, M. Schlosser and H. Garcia-Molina , "The EigenRep Algorithm for Reputation Management in P2P Networks". In Proceedings of WWW, Budapest, Hungary, 2003.

[19] Y. Yang, Q. Feng, Y. Sun, and Y. Dai, "Reputation trap: An powerful attack on reputation system of file sharing p2p environment," in the 4th Int. Conf. on Security and Privacy in Communication Networks, 2008.

[20] M. Srivatsa, L. Xiong, and L. Liu, "Trustguard: countering vulnerabilities in reputation management for decentralized overlay networks," in Proc. of the 14th Int. Conf. on World Wide Web, May 2005

[21] Y. Yang, Y. L. Sun, S. Kay, and Q. Yang, "Defending online reputation systems against collaborative unfair raters through signal modeling and trust," in Proc. of the 24th ACM Symposium on Applied Computing, Mar 2009.

[22] Y. Liu and Y. Sun, "Anomaly detection in feedback-based reputation systems through temporal and correlation analysis," in Proc. of 2nd IEEE Int. Conf. on Social Computing, Aug 2010.

[23] L. Xiong , and L.Liu," PeerTrust: Supporting Reputation-Based Trust for Peer-to- Peer Electronic Communities", IEEE Transactions on Knowledge and Data Engineering, Vol. 16, No. 7, July 2004.

[24] Y. Zhang, H. Chen, and Z.Wu, "A social network-based trust model for the semantic web. In Autonomic and trusted computing", Springer, pp. 183192, Berlin 2006

[25] R. Chen, X. Zhao, L. Tang, J. Hu, and Z. Chen, "CuboidTrust: a global reputation-based trust model in peer-to-peer networks," Fourth Int. Conf. on autonomic and Trusted Computing, pp. 203-215, 2007

[26] W. Wang , G. Zeng and L. Yuan. "Ant-based reputation evidence distribution in P2P networks". 2006.

[27] F. Gomez , G. Martinez and A. F. Skarmeta. "Tacs, a trust model for P2P networks". 2007 ; Springer.

[28] Y. Wang , V. Cahill , E. Gray , C. Harris and L. Liao. "Bayesian network based trust management". 2006 , Springer.

[29] T. Zhuo , L. Zhengding and L. Kai. "Time-based dynamic trust model using ant colony algorithm". 2006.

[30] A. Tajeddine ; A. Kayssi ; A. Chehab and H. Artail. "Patrol-f - a comprehensive reputation based trust model with fuzzy subsystems". 2006 ; Springer.

[31] F. Yu , H. Zhang , F. Yan and S. Gao. "An improved global trust value computing method in P2P system". 2006 ; Springer.

[32] E. Damiani, S.D. Capitani, S. Paraboschi, and P. Samarati. Managing and Sharing Servants Reputations in P2P Systems. IEEE Trans. Knowledge and Data Engineering, 2003, Vol. 15, No. 4, pp. 840-854,

[33] A.Das, M.M.Islam. SecuredTrust: A Dynamic Trust Computation Model for Secured Communication in Multiagent Systems. IEEE Transacions on Dependable and Secure Computing, 2012, Vol. 9, No. 2, pp.261-275.

[34] G.Silaghi and al., "Reputation-based Trust management systems and their applicability to Grids (Technical report), Network of Excellence 2007.

[35] H. Liu and J. Shen, "A Mission-Aware Behavior Trust Model for Grid Computing Systems", Int'l workshop on Grid and Cooperative Computing (GCC2002), Sanya, China, Dec. 26, 2002.

[36] S. Song and K. Hwang, "Fuzzy trust integration for security enforcement in grid computing," in International Symposium on Network and Parallel Computing (NPC2004), March. 2004.

[37] Wenjuan Li, Lingdi Ping, and Xuezeng Pan, "Use trust management module to achieve effective security mechanisms in cloud environment," in International Conference on Electronics and Information Engineering (ICEIE), vol. 1, Kyoto, Japan, pp. 14-19, 2010

[38] P.D. Manuel, T. Selve, and M.I. Abd-EI Barr, "Trust management system for grid and cloud Resources," in First International Conference on Advanced Computing (ICAC 2009), Chennai. India, pp. 176-181, 2009

[39] W.Li, and L.Ping, "Trust Model to Enhance Security and Interoperability of Cloud Environment", CloudCom 2009

[40] C. Lin, V. Varadhrajan, Y. Wang, and V. Pruthi, "Enhancing Grid Security with Trust Management", IEEE International Conference On Services Computing, SCC 2004

[41] A.Bradai and H.Afifi, "Enforcing Trust-based Intrusion Detection in Cloud Computing Using Algebraic Methods", IEEE International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2012.

[42] L. Lamport, R. Shostak , and M. Pease, "The Byzantine Generals Problem", ACM Transactions on Programming Languages and Systems, pp. 382-401, July.1982

[43] Mohit Agrawal, "The Perron-Frobenius Theorem and Google's PageRank Algorithm", 2010

[44] P.Mell, and T.Grance, "The NIST Definition of Cloud Computing", 2009

[45] Trend micro, "Cloud security survey", Global executive summary, June 2011

[46] V.D.Cunsolo, S.Distefano , A.Puliafito , M.Scarpa , "Volunteer Computing and Desktop Cloud: The Cloud@Home Paradigm", Eighth IEEE International Symposium on Network Computing and Applications, 2009.

[47] Abhishek Chandra and Jon Weissman, "Nebulas: Using Distributed Voluntary Resources to Build Clouds", HotCloud'09 Proceedings of conference on Hot topics in cloud computing, 2009.

[48] F.Azzedin, and M.Maheswaran, "Towards Trust-Aware Resource Management in Grid Computing Systems", 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID02), 2002.

[49] K.-J. Lin et al., "Reputation and Trust Management Broker Framework for Web Applications", Proc. IEEE Conf. e-Technology, e-Commerce, and e-Services, IEEE CS Press, pp. 262-269, 2005.

[50] Francesco Ricci and Lior Rokach and Bracha Shapira, "Introduction to Recommender Systems Handbook", Recommender Systems Handbook, Springer, pp. 1-35, 2011.

[51] D. Zeng. "Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering". ACM Trans. Inf. Syst., 22(1):116142, 2004.

[52] P.Zhang and al. "Survey of trust management on various networks", International Conference on Complex Intelligent and Software Intensive Systems, 2011.

[53] Marsh, S., "Formalising Trust as a Computational Concept. PhD thesis", University of Stirling, Department of Computer Science and Mathematics, 1994.

[54] A. Mejia et al., "A game theoretic trust model for on-line distributed evolution of cooperation in MANETs" Elsevier Journal of Network and Computer Applications.34:39-51, 01/2011.

[55] S. Ganeriwal, L.K. Balzano, and M.B. Srivastava, "Reputation-based framework for high integrity sensor networks," ACM Transitions on Sensor Network, vol. 4, no. 3, May 2008.

[56] K. Liu, N. Abu-Ghazaleh, and K.-D. Kang, "Location verification and trust management for resilient geographic routing," J. Parallel and Distributed Computing, vol. 67, no. 2, pp. 215-28, 2007.

[57] R.A. Shaikh, H. Jameel, B.J. d'Auriol, H. Lee, S. Lee, and Y.J. Song, "Group-based trust management scheme for clustered wireless sensor networks," IEEE Transactions on Parallel and Distributed Systems, vol. 20, no. 11, pp. 1698-1712, Nov.2009.

[58] T.Kim ,and H.Seo, "A trust model using fuzzy logic in wireless sensor network", World academy of science, engineering and technology, 42, 2008.

[59] Y. Zhang, H. Chen, and Z.Wu, "A social network-based trust model for the semantic web. In Autonomic and trusted computing", Springer, pp. 183192, Berlin 2006:

[60] L. Xiong , and L.Liu," PeerTrust: Supporting Reputation-Based Trust for Peer-to- Peer Electronic Communities", IEEE Transactions on Knowledge and Data Engineering, Vol. 16, No. 7, July 2004.

[61] S. Kamvar, M. Schlosser and H. Garcia-Molina , "The EigenRep Algorithm for Reputation Management in P2P Networks". In Proceedings of WWW, Budapest, Hungary, 2003.

[62] R. Chen, X. Zhao, L. Tang, J. Hu, and Z. Chen, "CuboidTrust: a global reputation-based trust model in peer-to-peer networks," Fourth Int. Conf. on autonomic and Trusted Computing, pp. 203-215, 2007

[63] G.Silaghi and al., "Reputation-based Trust management systems and their applicability to Grids (Technical report), Network of Excellence 2007.

[64] H. Liu and J. Shen, "A Mission-Aware Behavior Trust Model for Grid Computing Systems", Int'l workshop on Grid and Cooperative Computing (GCC2002), Sanya, China, Dec. 26, 2002.

[65] S. Song and K. Hwang, "Fuzzy trust integration for security enforcement in grid computing," in International Symposium on Network and Parallel Computing (NPC2004), March. 2004.

[66] Wenjuan Li, Lingdi Ping, and Xuezeng Pan, "Use trust management module to achieve effective security mechanisms in cloud environment," in International Conference on Electronics and Information Engineering (ICEIE), vol. 1, Kyoto, Japan, pp. 14-19, 2010

[67] P.D. Manuel, T. Selve, and M.I. Abd-EI Barr, "Trust management system for grid and cloud Resources," in First International Conference on Advanced Computing (ICAC 2009), Chennai. India, pp. 176-181, 2009

[68] W.Li, and L.Ping, "Trust Model to Enhance Security and Interoperability of Cloud Environment", CloudCom 2009

[69] C. Lin, V. Varadhrajan, Y. Wang, and V. Pruthi, "Enhancing Grid Security with Trust Management",IEEE International Conference On Services Computing, SCC 2004

[70] A.Bradai and H.Afifi, "Enforcing Trust-based Intrusion Detection in Cloud Computing Using Algebraic Methods", IEEE International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2012.

[71] L. Lamport, R. Shostak , and M. Pease, "The Byzantine Generals Problem", ACM Transactions on Programming Languages and Systems, pp. 382-401, July.1982

[72] Mohit Agrawal, "The Perron-Frobenius Theorem and Google's PageRank Algorithm", 2010

[73] A.Bradai and H.Afifi, "Enforcing Trust-based Intrusion Detection in Cloud Computing Using Algebraic Methods", IEEE International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2012

[74] Cloud security survey, Global executive summary, Trend micro, June 2011

[75] X. Jing, "A brief survey on the security Model of Cloud Computing ", ISDCAB 2010.

[76] B.Kandukuri, " Cloud Security Issues", 2009 IEEE

[77] H.Takabi, " Security and privacy challenges in cloud computing environments " 2010 IEEE

[78] S. Ganeriwal, L.K. Balzano, and M.B. Srivastava, "Reputation-based framework for high integrity sensor networks," ACM Transitions on Sensor Network, vol. 4, no. 3, May 2008.

[79] K. Liu, N. Abu-Ghazaleh, and K.-D. Kang, "Location verification and trust management for resilient geographic routing," J. Parallel and Distributed Computing, vol. 67, no. 2, 2007, pp. 215-28.

[80] R.A. Shaikh, H. Jameel, B.J. d'Auriol, H. Lee, S. Lee, and Y.J. Song, "Group-based trust management scheme for clustered wireless sensor networks," IEEE Transactions on Parallel and Distributed Systems, vol. 20, no. 11, Nov. 2009, pp. 1698-1712.

[81] Marsh, S., "Formalising Trust as a Computational Concept. PhD thesis", University of Stirling, Department of Computer Science and Mathematics, 1994.

[82] A. Mejia, N. Peña, J. Muñoz, O. Esparza, M. Alzate, "A game theoretic trust model for on-line distributed evolution of cooperation in MANETs" Elsevier Journal of Network and Computer Applications. 01/2011; 34:39-51

[83] T.Kim and H.Seo "A trust model using fuzzy logic in wireless sensor network" World academy of science, engineering and technology, 42, 2008.

[84] M. Castro and B. Liskov, "Practcal byzantine fault tolerance", In Proc.OF OSDI, New orleans, LA, 1999.

[85] L.Vokorokos and A.Balaz, "Host-based Intrusion Detection System",Intelligent Engineering Systems (INES), 14th International Conference, 2010

[86] S.Roschke, F.Cheng, C.Meinel, "Intrusion Detection in the Cloud", IEEE 2010

[87] Chi-Chun Lo, Chun-Chieh Huang and Joy Ku, "A Cooperative Intrusion Detection System Framework for Cloud Computing Networks",39th International Conference on Parallel Processing Workshops, 2010

[88] Chunsheng Li, Qingfeng Song, Chengqi Zhang: "MA-IDS Architecture for Distributed Intrusion Detection using Mobile Agents", Proceedings of the 2nd International Conference on Information Technology for Application (ICITA), 2004.

[89] F. Wang, C. Huang, J. Zhang, and C. Rong, "IDMTM: A novel intrusion detection mechanism based on trust model for ad hoc networks," 22nd IEEE International Conference on Advanced Information Networking and Applications, 2008, pp. 978-84.

[90] P. Ebinger and N. Bibmeyer, "TEREC: Trust evaluation and reputation exchange for cooperative intrusion detection in MANETs," 7th Annual Comm. Networks and Services Research Conf. 2009, pp. 378-385

[91] F.Bao, I.Chen, M.Chang and J.Cho, "Trust-based intrusion detection in wireless sensor networks",IEEE International Conference on Communications (ICC 2011), Kyoto, Japan, June 2011.

[92] Mohit Agrawal, The Perron-Frobenius Theorem and Google's PageRank Algorithm, 2010

[93] M. Dirani and T.chahed, "Framework for resource allocation in heterogenous wireless networks using game theory", in EURO-NGI, Berlin, Heidelberg: Spring-Verlag, pp. 144-154

[94] Abe hirota, o.yoshihiro, o.mizuki, k.kazuhiko, "Optimization of Intrusion Detection System Based on Static Analyses", JST , Japan 2004

[95] A Hofmann, B Sick, "Evolutionary optimization of radial basis function networks for intrusion detection ",Neural Networks, 2003.

[96] Lye K, Wing J. "Game strategies in network security", In: Proceedings of the Foundations of computer security workshop. Copenhagen, Denmark, July 2002

[97] Alpcan T, Basar T. "A game theoretic approach to decision and analysis in network intrusion detection". In: Proc. of the 42rd IEEE conference on decision and control (CDC), Maki, HI; . p. 2595 600, December 2003

[98] F. Wang, C. Huang, J. Zhang, and C. Rong, "IDMTM: A novel intrusion detection mechanism based on trust model for ad hoc networks," 22nd IEEE International Conference on Advanced Information Networking and Applications, pp. 978-84, 2008

[99] P. Ebinger and N. Bibmeyer, "TEREC: Trust evaluation and reputation exchange for cooperative intrusion detection in MANETs," 7th Annual Comm. Networks and Services Research Conf , pp. 378-385, 2009

[100] L.chen and J.Leneutre, "A game theoretical framework on intrusion detection in heterogenous networks ", Information Forensics and Security, 2009

[101] T. Alpcan and T. Basar, "An intrusion detection game with limited observations", in 12th Int. Symp. on Dynamic Games and Applications, Sophia Antipolis, France, July 2006

[102] S.Liu and al, " A game theoric approach to optimize the performance of host based IDS", Networking and Communications, . WIMOB '08. IEEE International Conference on Wireless and Mobile Computing, 2008

[103] Agah A., Das S.K., Basu K., Asadi, M. "Intrusion detection in sensor networks: a non-cooperative game approach "In: Proceedings of the Third IEEE International Symposium on Network Computing and Applications, 2004

[104] H.Otrok and al. " A game-theoretic intrusion detection model for mobile ad hoc network ", Computer Communications 31 708721, 2008

[105] P. Paruchuri, J. P. Pearce, J. Marecki, M. Tambe, F. Ordonez, and S. Kraus. "Playing games with security: An efficient exact algorithm for Bayesian Stackelberg games". In AAMAS-08, pages 895-902, 2008

[106] C. Kiekintveld, M. Jain, J. Tsai, J. Pita, M. Tambe, an F. Ordonez." Computing optimal randomized resource allocations for massive security games". In AAMAS, pages 689-696, 2009

[107] C.Kiekintveld, J.Marecki and M.Tambe, " Methods and algorithms for infinite bayesian stackelberg security games ", GameSec'10 Proceedings of the First international conference on Decision and game theory for security, 2010

[108] H.Wei and H.Sun " Using bayesian game model for intrusion detection in wireless adhoc networks ",Int. J. Communications, Network and System Sciences, 2010

[109] Osborne MJ. "An introduction to game theory". New York: Oxford University Press; 2004

[110] A.Bradai and H.Afifi, "Enforcing Trust-based Intrusion Detection in Cloud Computing Using Algebraic Methods", IEEE International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2012

[111] Shigen Shen,"A game-theoretic approach for optimizing intrusion detection strategy in WSNs", IEEE 2011

[112] Mohit Agrawal, The Perron-Frobenius Theorem and Google's PageRank Algorithm, 2010

[113] S. Kamvar, M. Schlosser and H. Garcia-Molina . "The EigenRep Algorithm for Reputation Management in P2P Networks". In Proceedings of WWW, Budapest, Hungary, 2003.

[114] J.F. Nash, Jr., "Non-cooperative games", Annals of Math., 54(2):286295, September 1951

[115] http://en.wikipedia.org/wiki/Pareto$_e ffficient$

[116] R. Zhou, K. Hwang, and M. Cai. "Gossiptrust for fast reputation aggregation in peer-to-peer networks". IEEE Trans. on Knowl. and Data Eng., 20(9), 1282-1295, 2008.

[117] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah Randomized gossip algorithms, IEEE Transactions on Information Theory, Special issue of IEEE Transactions on Information Theory and IEEE ACM Transactions on Networking, June 2006, 52(6):2508-2530.

[118] D. Kempe, A. Dobra, and J. Gehrke, Gossip-Based Computation of Aggregate Information, Proc. 44th Ann. IEEE Symp. Foundations of Computer Science (FOCS03), Oct. 2003.

[119] R. Zhou " Gossip-Based Reputation Management for Unstructured Peer-to-Peer Networks "

[120] Vincenza Carchiolo, Alessandro Longheu, Michele Malgeri, Giuseppe Mangioni, " The effect of malicious peers in a gossip-based reputation system " 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human 2009, Seoul, Korea, 24-26 November 2009

[121] A. M. Frieze and G. R. Grimmett. The shortest-path problem for graphs with random arc-lengths. Discrete Applied Mathematics, 10:57 77, 1985.

[122] Ariel D. Procaccia and Yoram Bachrach and Jeffrey S. Rosenschein " Gossip-Based Aggregation of Trust in Decentralized Reputation Systems " IJCAI 2007

[123] D. Kempe, A. Dobra and J. Gehrke, "Gossip-based Computation of Aggregate Information", Proc. of IEEE Symposium on Foundations of Computer Science, Cambridge, MA, Oct.2003.

[124] http://www.mathworks.com/matlabcentral/fileexchange/11947-b-a-scale-free-network-generation-and-visualization