PhD Dissertation

International Doctorate School in Information and Communication Technologies

DISI - University of Trento

Application of Reputation Management Systems in Autonomic Communication Networks

Roberto G. Cascella

Advisor:

Prof. Roberto Battiti

Università degli Studi di Trento

November 2007

to my parents

Abstract

The increasing use of wireless infrastructure and portable technologies, such as laptops, cellular phones, and Personal Digital Assistants (PDAs), has enabled nodes to connect to wireless networks whenever access is available or to form spontaneous ad hoc networks to exchange files or context information.
This new communication paradigm changes the role of the user from consumer to content provider and it affects the way self-organizing systems, like peer to peer networks, are used. Furthermore, device mobility and user mobility create new interesting services designed specifically for nomadic users.
Peer to peer networks need to have self-organization properties because of the lack of a centralized authority. This implies that nodes should self-manage and cooperate to sustain the availability of the resources in the system. In this context, researchers have suggested the use of reputation management schemes to foster cooperation and to mitigate the attacks of misbehaving nodes.
In this thesis, we analyze the principles of autonomic communication system to define new security requirements that targets malicious and selfish nodes. We evaluate the basic components of reputation management systems and study the cost vs. the benefit of using reputation in peer to peer applications.
Borrowing techniques from game theory, we discuss the importance for a node to build and use its reputation value. We propose a framework, based on the generalized form of the Iterated Prisoner's Dilemma, to model the interactions of rational and selfish nodes in distributed systems. We study how a node takes into account the change of its reputation when deciding its behaviour in a trans-

action and discuss the Nash equilibrium in the system.
Then, we estimate the performance of reputation management schemes for specific applications. We simulate the capability of these schemes in reducing inauthentic and corrupted file transfers in end-user collaborative content-distribution systems.
Finally we define a new mechanism to improve the application of reputation management schemes in mobile scenarios. The token-based approach enables the correlation of reputation values earned in different contexts and virtual communities. This results in reducing the problem of reputation bootstrap in new communities and in accelerating the identification of malicious nodes.

Keywords
Autonomic systems, Reputation management, Game theory, Mobility

Acknowledgments

First of all I would like to thank my family, not only my parents, but this time also my sisters and my brothers in law for their unending support during my studies and my little nephew for *attending* the defense. They have been all the time present whenever I need.

A special thank also to Elisa who has been part of my life during the last period of my thesis. She has shared with me the stress, always encouraged me to pursue my objectives, and contributed to the happiness that took me to the end of my studies. I acknowledge also Clarisse who supported my choice of coming in Trento. Finally, I am really grateful to my friends and all the people who I have meet during my stay in Trento and who have *enjoyed* my presence during these years.

It is now the turn of the people with whom I have worked during these years, without their help and guidance I would have not accomplished the work presented in this thesis. I would like to express my gratitude to Roberto Battiti, my advisor, who gave me a lot of hints and advised me for the best for my research and my work. I have learnt a lot from him not only in the research field.

I was also very fortunate to work in the NetMob and LION groups where I have met people with whom I enjoyed discussions and who where all the time ready to help me and give me their support. A special thank to Danilo Severina, who knows all the plots in this thesis better than I do, to Sanajy, who proofread an early version of this manuscript, to Anurag Garg, with whom I have started to work when I came in Trento, and to Damiano Carra, with whom I have started the PhD studies and discussed many ideas.

I also acknowledge Mauro Brunato, Renato Lo Cigno, Bruno Crispo and Fabio Massacci for their advises and insights, Miklos Telek, with whom I discussed part of my work, for his feedback and constructive criticism. I would also thank the members of my final exam committee, Ioannis Stavrakakis, Michael Welzl and Pietro Michiardi, who have given me feedback and useful comments for my future research.

Even if he was not involved in my research activities in these last years, I am really grateful to Professor Gerald Maguire Jr. who has initiated me in the research field during my Master Thesis and who has shown me devotion to scientific research. He was all the time an example for pursuing my studies.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

In the last decade the communication framework is shifting from a client-server model to a client-centric one. This change, driven by the increased number of home consumer devices and the need of the users to obtain media-content or to participate in online communities like auction sites has led to the growth of a new type of communication scheme that does not always involve a centralized authority. Peer-to-peer networks, either mobile or fixed, and distributed systems in general, are the response to the new needs of the consumers.

In this context, the role of the user changes from consumer only to provider and consumer at the same time. Thus, networks are populated by the presence of new services and content directly offered by the users which communicate without the need of a central authority. The formation of these so called *virtual communities* brings together people who are completely strangers operating behind their electronic devices without the need to reveal their real identities. In addition, the widespread usage of mobile devices and the presence of wireless connectivity enables the creation of mobile virtual communities that are more dynamic, thus, increasing the difficulty in establish relationship among strangers.

In this scenario, automatic reconfigurable systems assume a key role to respond to the user's needs for a highly scalable and dynamic system. This system must efficiently react to network changes and adapt to the context of the mobile or fixed communication. Users participate in the system functions actively as they provide services but the reason behind their contribution is the benefit they obtain from interactions with other nodes.

However, putting intelligence in the network itself and making the users responsible for the survivability of the system create new challenges as distributed systems are more vulnerable to attacks of malicious nodes that want to disrupt the network. In this scenario, the system itself should be able to react to possible failures and automatically reconfigure itself to eliminate malicious users without requiring the intervention of a centralized authority.

1.2 Introduction to the problem area

The growing attention to distributed systems is justified by the inherent properties of these networks. Scalability and distribution of the load from a central server to multiple nodes are the most important features that have characterized their wide adoption. Distributed grid computing and file-sharing system like Napster are the first attempts to exploit the end-point processing power or available content.

The cost of the infrastructure is shared among the users that participate in the system activities. In addition, the user can create media-content or sell the computational capabilities of its personal device to perform complex tasks. More recently, online-auction sites like eBay have gained popularity; they allow every user to advertise goods and bid for available items. However, the common feature of these systems is the presence of a central authority, which guarantees the execution of the protocol and monitors the activity of every node, which needs to register before joining the system.

In dynamic and mobile environments where nodes form short-lived and online virtual communities the presence of a central server to store bindings between resources and locations is impractical. In particular, this is true when the community is limited in time and it is created for a specific scope. Due to the dynamic changes and due to the fact that nodes are *strangers* and belong to different organizations we cannot rely on a trusted central point that can monitor activities and decide in case of disputes.

Fully autonomic peer-to-peer systems are the response to the users' needs. In these systems, users interact **directly** with each other to perform a service, to exchange data, to share files, computational power, and other resources. An attempt to build a fully decentralized network is represented by *content distribution networks* (CDNs) which consist of globally distributed servers located strategically at the edge of the Internet. Service providers, such as web sites, rely on these systems to distribute popular information in close proximity to the user. For this reason, CDNs prove to be effective for managing large amount of traffic in the Internet.

This need to decentralize the functions of the system to end-points and to maintain survivability of the system itself is the premise for the evolution of autonomic communication systems. The intelligence and the monitoring capabilities required by a system to function properly are shared by the nodes forming these networks. Thus, the need for a centralized authority has been substituted by a system which is capable of self-organizing and self-preserving.

Autonomic systems aim at incorporating methods to monitor their dynamic behaviour and to react in automated ways in order to effectively reconfigure themselves to adapt to new environments. These concepts of self-management and self-adaptation, as defined in [65], lead to a system that should be capable of self-configuration, self-healing, self-optimization, and self-protection.

This new system paradigm is fully applicable to all decentralized peer to peer systems, like content distribution networks, where **cooperation** among end-users is the common characteristic. For instance, collaborative end-user content-distribution systems are cost-effective, inherently scalable and more resilient to network and equipment failures. As the server bandwidth is limited, the idle bandwidth available between the users is used to deliver content efficiently. This system also autonomically self-adapts to the users' demand and to changes in the network such as node departures and arrivals.

In this dynamic network environment, more content is available, and the methods incorporated into the system must guarantee self-scalability and lead to a self-organization and self-management of the resources. Nevertheless, the issue that remains to be addressed is how to enforce cooperation and how to monitor the behaviour of the nodes in autonomic communication systems.

1.3 Motivation of the thesis

Most of the effort in autonomic communication systems is focused on the definition of new paradigms to adapt the system properties to network changes or to the context of the communication. This new paradigm must enable the formation of autonomic *virtual communities*. Autonomic communication systems are formed by strangers and heterogeneous components which might be under different administrative domains. These entities interact to achieve their specific goals and to sustain the scalability of the system. However, the difficulty in establishing direct relationships complicates the measurement of nodes' trustworthiness.

In a collaborative network, the participation of the nodes is strongly dependant on whether, and how much, they benefit from it. Self-aware nodes may act strategically and try to improve their local utility by not participating in transactions that are not of interest to them. For instance, a node falsely declares to be uploading and serving content to other peers, so as to be able to benefit from new content in exchange for its services. Furthermore, malicious users may introduce fake or mislabelled content in the network making the use of the service problematic if not dangerous.

The participation of the nodes and the quality of the communication cannot be granted in a systems formed by *strangers*. If the survivability of peer-to-peer networks relies on the nodes' willingness to fulfil their obligations, the system might not function properly: non-cooperative or selfish behaviour will be privileged and predominant. This occurs because individual rational nodes are incentivized to maximize their own use of the resource while the costs of the service are shared between all those to whom the service is available. As a result of this *tragedy of the commons* [60], selfish nodes do not share their resources if they cannot increase their *utility*.

In un-managed and fully distributed systems where there is no authority to force nodes to follow an altruistic behaviour (where nodes cooperate without reward), incentive mechanisms leading to *reciprocal altruism* are required to foster cooperation among nodes. By definition, reciprocal altruism, or *direct reciprocity* [91], exists when an entity provides a service to another entity without expecting any immediate payment or compensation. This cooperative action must result in increasing the benefit for the receiver of the service and the benefit must be larger then the cost the provider has. Reciprocal altruism is only achieved when also the beneficiary of the service is cooperative in future inverse situations.

In social and economic science numerous works study entities interactions in virtual communities and discuss the impact of reciprocate behaviour of the nodes in their interaction [41, 77]. Cooperation upon previous successful experience is applicable if the same nodes interact frequently during their lifetime. However, peer-to-peer systems or large distributed communities face the problem of nodes that sporadically meet. Thus, these schemes are not effective in this context due to the limited number of direct interactions. Another approach is based on indirect collaboration of nodes. This mechanism implies that *indirect reciprocity* relationships can be established if the transactions are monitored and the result of this observation is shared among the nodes [77].

Indirect reciprocity is an extension of the concept of reciprocal altruism as it is based on the observation of actions between two entities that meet randomly. In case of indirect reciprocity, these two entities do not need to meet again as the behaviour in the interaction is observed by a subset of the population that will inform the others on the outcome. Thus, the application of indirect reciprocity leads to the definition of reputation management schemes. Reputation management systems measure trustworthiness in a collaborative-based network. They monitor participants' behaviour in all transactions to compute the nodes' trustworthiness. Then, this information is used to build knowledge of the behaviour of the nodes to estimate the expected quality of the nodes in providing resources. Thus, a reputation mechanism needs to be installed to provide the necessary trust required for fostering cooperation in autonomic distributed systems.

1.4 Contribution of the thesis

This thesis tackles the problem of cooperation in autonomic communication systems by motivating the need for cooperation and by analyzing threats and available solutions in the literature. This study enables the definition of the building blocks of a reputation management system.

The specific objective of a reputation management system is to facilitate nodes to decide whom to trust for providing the correct quality of services or resources to the requesting node. To serve this objective a reputation management system must perform three distinct functions: 1) collection, 2) aggregation and 3) dissemination of trust information. Moreover, the reputation management system must *motivate* nodes to show cooperative behaviour and discourage malicious nodes to take an active role in the network.

The enforcement of reputation in autonomic systems has a cost in terms of messages required to disseminate feedbacks and to synchronize reputation values. In this thesis we consider two approaches to collect and disseminate data: proactive and reactive schemes. These methods are the two possible extreme cases and other hybrid approaches can be easily derived by integrating them. We analyze the impact of these collection/dissemination approaches and derive conclusions to quantify the impact of overhead messages on the system.

The analysis is carried out by using a simplified model of a reputation management scheme. In fact, we are interested in studying the impact of the adoption of a general reputation scheme in distributed systems. The conclusions can be extended to any reputation management system and to specific application contexts.

The results of the analysis show the importance of reputation systems in fully un-managed networks. We can also determine which is the best approach based on the context of the application and on the frequency of the interactions between the nodes. The collection and dissemination frequency can be adjusted to respond to the specific needs. These results are general and they can be applied to content distribution networks as well as to any dedicated peer-to-peer system that implements a reputation management scheme.

1.4.1 Modelling reputation

This thesis gives a theoretical evidence of the need to implement a reputation management system. We develop a model to study how *building* reputation is important for a node's future interactions or how a node *values* its reputation.

Previous work assumes reputation as a metric to define a *cooperation strategy* or to implement a differential service incentive scheme. This is not sufficient to explain why a node should increase its reputation value and to reason on the adoption of reputation management systems from the nodes point of view. If we consider rational nodes, that strategize to increase their expected utility from the system, there is no clear understanding of the role of reputation in selecting a specific action/strategy. In most cases nodes want to collaborate to keep their trust value above a certain threshold that allows them to consume system resources at the minimum contribution level.

Reputation systems are at the boundary between social and economic sciences as well as evolutionary biology and computer science. For this reason, we closely analyze the adoption of reputation management systems from an interdisciplinary perspective. In our formalization of the system properties, we discuss the evolution of the system under the application of a reputation management scheme.

We focus on the issue of reputation and trust building in a decentralized system. Moreover, we exploit specific economic theories to model peer-to-peer systems. In particular, we discuss what the need for cooperation is in an autonomous distributed system formed by selfish agents and we focus the attention on reputation with respect to its definition of future revenue, i.e. the granted permission to access resources in future.

We propose a game theoretic approach based on the generalized form of the *iterated prisoner's dilemma* to study reputation management systems. We derive the *Nash equilibrium* for the reputation game and we show that reputation is sufficient to sustain cooperation. A quantitative analysis of the time a node must wait before being rewarded for its cooperative behaviour is given and it is based on the reputation level of the nodes.

Simulations show the impact of different strategies selected by nodes on the system performance. The performance is measured in terms of cooperative transaction. The probability distribution of the reputation value in the system is derived to understand how reputation systems model the behaviour of a node.

We also assess the performance of the reputation scheme in the presence of nodes that always defect when interacting with other entities. Results show that reputation is needed to isolate faulty components.

1.4.2 Application to content distribution networks

For the application of reputation systems to autonomic distributed systems, this thesis presents ideas of reputation management and their development and application for soft management in autonomic and collaborative content distribution systems. To measure the trustworthiness of a peer, a feedback-based trust management system is applied to the case of content distribution systems.

The aim is to incorporate methods to sustain self-optimization of the resources by limiting corrupted downloads and to provide a minimal degree of protection of the shared content. In this work, we extend an existing reputation management system to function in a collaborative content distribution system. Unlike other reputation management schemes, our approach targets collaborative content distribution through splitting files into blocks and uploading them to different peers followed by the exchange of these blocks among peers.

We define the type of malicious nodes that are active in content distribution systems and we simulate two scenarios: when the node(s) providing the fake blocks can be identified and when they cannot be identified. We then study in detail the latter case – which is a more accurate representation of systems currently in place – and measure the impact of several variables on the performance of the reputation management system.

Our solution to the problem of malicious corruption of blocks is to allow a peer to select its transaction partners based on the trust it places on them. A collaborative content distribution system is particularly susceptible to *pollution attacks* since a single bad block can render the entire download worthless.

The results show that a large percentage of malicious nodes is identified and corrupt block transfers are prevented if reputation management systems are used. The application of reputation systems increases the fraction of good content downloaded in both cases when the sender of the corrupt blocks can be identified and when he cannot be identified.

In addition, we evaluate the impact of the number of blocks and the percentage of malicious peers. As the number of blocks required increases, the probability that the source successfully downloads the content decreases. Hence, networks with a large number of malicious nodes must keep the number of blocks required to reconstruct a file as low as possible.

An immediate applicability of our work is to content networks relying on network coding. With network coding, blocks are re-coded at each intermediate node, and block verification cannot be done by using a predetermined hash of the block. Another application is on the popular file sharing BitTorrent: reputation trust metrics can correlate downloads of different files.

1.4.3 Mobility support for reputation

In this thesis we consider a mobility scenario, when nodes frequently join and leave virtual communities while they are on the move. The application of existing reputation management systems to mobile devices is not immediate. Nodes are considered as new entrants each time when they join a community and their past behaviour in other context or communities is not evaluated.

In this setting, nodes have an incentive to misbehave when they join a community for a short time as the reputation management system cannot immediately give an accurate estimation of the behaviour of the node. For this reason, most of the existing reputation management systems consider new nodes as malicious nodes and assign them a low reputation value. This practice is also used to avoid that a node joins a community with multiple identities to whitewash them from any bad reputation.

Even if few transactions are required to build an initial value, during this period the node cannot access services due to its low reputation, since service providers deny the access to their resources. We design a token-based mechanism to help the bootstrap of a new node in a community by exploiting its past transactions in other clusters. This mechanism extends existing reputation management systems to support mobility and enables the correlation of transactions in different contexts effectively.

The results show that the token-based mechanism improves the performance of reputation management systems in mobile applications. We also consider nodes that change behaviour when they join different clusters, i.e., *milk* their behaviour. The mechanism is able to detect the presence of malicious nodes and it effectively increases the fraction of cooperative transactions.

The token-based mechanism is general-purposes and it can be applied to any system that has multiple virtual communities. Nodes have the capability to join different communities created for a specific reason. The token can contextualize the reputation values obtained in other context and link the reputation values. Different weights can be associated to reputation values formed in communities based on the judging capability of the community or based on the scope of the community itself.

1.5 Assumptions for the research

There are some important issues in reputation management systems and their application to distributed systems that are not covered in this thesis.

Reputation System details are not provided in this thesis, as we study the application of a general reputation management system and we extend the functionality of an existing mechanism. We do not provide the design of a new reputation management system as we are interested in capturing the general properties of reputation systems and in studying how these systems are effective in autonomic communication systems.

Networking properties are not considered when studying the impact of the reputation management systems. In particular, the reputation model we formulate is abstracted from a specific context. This thesis assumes the presence of an overlay network and of routing schemes to propagate the messages. We do not deal with the formation of a network and how reputation management systems can be used to guide the construction of an overlay.

Security and Privacy is important to ensure integrity and non-repudiation of the opinions the node form on other nodes. We do not attempt to discuss the details of the implementation of cryptographic algorithms or how the keys are distributed among the nodes. We assume that nodes have a certificate and use public cryptography to digitally sign their evaluation.

Moreover, we do not design any specific application to guarantee unlinkability of the information. In particular, the information provided by the token-based mechanism can be exploited to track the movement of a node inside an area. Anonymity mechanisms can be used to provide untraceability of the location of users.

Identities are required to bind a reputation value to a node and to evaluate the transactions of a node. In this thesis, we do not propose any mechanism to generate identities and to guarantee that these identities are unique in the system. We assume the presence of a global identity and we assume that nodes cannot spoof identities and steal reputation values.

1.6 Outline of the thesis

The content of each chapter and the most important finding are briefly discussed in the following outline.

Chapter 2 introduces the concept of autonomic management systems and discusses the threats and specific requirement to construct a self-organized and self-managed system. We motivate the need for cooperation in autonomic systems and present the solutions to foster cooperation. The analysis is specific to the application of reputation and outlines the important features this mechanism must have to guarantee self-preservation of autonomic communication systems.

Chapter 3 details the individual components of reputation management systems and presents the metrics to evaluate the performance of reputation schemes. In Chapter 3, we critically discuss the existing reputation management systems. The ROCQ reputation management system is described in detail as it is the reference scheme that we extend to apply reputation in different context. This chapter also gives an overview of the security issues in reputation management systems.

Chapter 4 studies the extra signalling these schemes introduce in an overlay network and we derive conclusions on the accuracy of the reputation system compared to the communication overhead.

Chapter 5 introduces the *reputation game* that we use to formalize an autonomic system. With this game we are able to justify the use of reputation management systems.

Chapter 6 presents an application of the reputation management system to content distribution networks. We analyze the impact of different types of malicious nodes and evaluate the performance.

Chapter 7 extends reputation management systems to support mobility. A token-based mechanism is proposed and the performance of the system under different mobility scenarios is evaluated.

Chapter 8 summarizes the results achieved in this thesis and gives guidelines for future work.

Chapter 2

An autonomic approach to cooperation

2.1 Autonomic systems: a definition

Autonomic systems are systems capable of operating and managing in an autonomous way without requiring external intervention. The core of an autonomic system consists of its capability to observe the working environment and the context of the communication. Given the task that must be accomplished and the result of the observation, the system adapts and reconfigures to serve its purposes. These concepts of self-management and self-adaptation, as defined in [65], define a system that should be capable of self-configuration, self-healing, self-optimization, and self-protection.

Self-configuration means that new components and software can be added to the system. The system must respond adaptively to changes in the environment with no disruption of the services provided. The system is termed *self-healing* when it is resilient to malfunctioning by detecting and isolating failed components. By monitoring the resources, a system must *self-optimize* their usage so that resources can be provisioned to meet the user needs without requiring human intervention. Moreover, the system must anticipate attacks from malicious users and protect against unauthorized access to the resources (*self-protection*).

The autonomic concepts are applicable to many computing fields. In particular, the rapid growth of the number of portable devices has opened an interesting research area that uses the autonomic principles to design an autonomic network. This autonomic network must deal with the complexity of managing a huge infrastructure sustained by available pervasive devices that sense the environment to report context information.

Mobile ad-hoc and peer-to-peer networks are two important examples of the need for the deployment of autonomic systems. These systems are characterized by heterogeneous entities with different computational capabilities. In this scenario, the systems must enable devices to propagate information in the network, and at the same time, to respond autonomically to network changes and to sustain the scalability of the system. These tasks are difficult to achieve, mainly because nodes can be on the move and they can exploit services that are contextualized to the environment [39].

2.2 Security paradigms in autonomic systems

Different kind of services and applications can be deployed in an autonomic communication system, but the common feature of these applications is the absence of a centralized authority. Federations of heterogeneous nodes self-organize to form an open environment where nodes dynamically move and self-adapt while the system changes. In the system, the lack of a trusted third party, that bootstraps the trust in the system, does not allow to establish *a priori* trust relationship among entities.

In this open environment, the application of autonomic principles requires dynamic paradigms for service composition and mechanisms to control the access to resources. Nodes actively participate in the process of service creation and are no longer within the boundary of a secure and trusted domain. In this scenario, strangers negotiate access to services and new security measures must be deployed to take into account changes in the services and in the system itself. In autonomic communication, the role of the trusted third party is shared among the users, who must monitor the network operation to self-preserve the system.

Mechanisms that respond to malicious and intentional attacks on the network infrastructure must cope with the distributed and dynamic nature of the system. These security mechanisms must monitor the system and proactively detect and identify threats to automatically fight or to activate reactive defensive countermeasures. Thus, autonomic systems require self-preservation capabilities that go beyond traditional access and authorization control.

2.2.1 Trust relationship and access control

The traditional basic security services to provide confidentiality, integrity, authentication and non-repudiation must be complemented by new security principles that are specific to autonomic communication systems. Security must provide an infrastructure to enable the formation of trust relationship in an open environment and to provide controlled access to resources. Other concepts like privacy and anonymity can be associated to the need of security in particular application scenarios. The need for privacy and anonymity might come from nodes that are willing to contribute to the system but do not want to disclose personal and sensible information, such as location, interactions and so on.

To establish security relationships, suitable key management and distribution protocols must be deployed. Traditional schemes rely on centralized authority for the verification and distribution of keys and, therefore, are not adequate for autonomic communication systems. New approaches based on the concept of the *web of trust* of Pretty Good Privacy (PGP) have been proposed. The certificates are stored in a completely distributed fashion and trust between two users is established on demand by merging the two local certificate repositories to find a match. This approach cannot work properly in a dynamic system because the multiple key pairs can be generated and its success depends on the way the local repository is constructed.

However, other self defending mechanisms that respond dynamically to malicious attacks can be incorporated in the security infrastructure. Nodes can be authorized to access resources but they can also provide poor services to the system community or in the worst case deny holding content to spare their resources. These attacks are outside the normal operation of the system and they can be difficult to detect.

2.2.2 Malicious behaviour control

Schemes to tackle malicious activities of the nodes require a detailed knowledge of the entire system and monitoring capabilities to timely detect possible threats. The detection and the monitoring must be implemented in a distributed fashion to follow the specification of autonomic systems and to share the cost of this activity among the members of a federation. In particular, these monitoring and *reactive* schemes must deal with abuse of resources, deny of possession of a resource, and false or poor quality service provision.

Although trust relationship could be efficiently deployed in an open environment, the survivability and self-preservation of the system resides in the contribution level of each single entity. False or corrupted content can reduce drastically the performance of an autonomic system. In particular, these systems self-adapt to the context of communication in a feedback control loop fashion by exploiting the information provided by the nodes.

Traditional cryptographic, or *hard security*, mechanisms are not able to protect the system from malicious sources of information. This occurs because *hard security* protects the system from nodes which try to steal or modify the information. As a consequence, the lack of a coordination point typical of a centralized entity necessitates the definition of an incentive mechanism to foster the correct behaviour and cooperation.

Another important new security paradigm introduced by autonomic communication systems is *cooperation*. From a security perspective, incentive mechanisms, that deal with cooperation, introduce a new dimension in the common view of security principle which can be classified as *soft-security*, or *social control mechanisms*.

In conclusion, security solutions in autonomic communication system must be deployed to protect the system from traditional *active and passive* attacks and from a new class of attack called *misbehaviour*. It is important to mention that the use of only *soft security* mechanisms does not protect the system from attacks, but these mechanisms are a important complement to traditional cryptography protection. In the following part of this thesis we study and propose countermeasures to mitigate the effect of malicious nodes inside the system and to motivate entities for cooperation.

2.3 Threats to autonomic communication systems

This section discusses the attacks that can be made targeting an autonomic communication system. We limit the analysis to attacks to the system caused by misbehaving nodes. The objective of this section is not necessarily to provide solutions to such attacks but instead to highlight the new attack scenarios in autonomic communication systems.

2.3.1 Adversaries: selfish and malicious nodes

Self-organized systems are composed by heterogeneous components that might be under control of different organizations, with different goals in mind. In many cases they can act selfishly and do not participate in the operations defined in the communication protocol or they can act maliciously by injecting false data in the network. These two types of entities are the adversaries as they want to harm other peers or the system as a whole.

In particular, nodes might act selfishly in the sense that they do not share the content they own or the data they have gathered from other nodes in previous interactions. For instance, mobile or sensor nodes are battery powered and have limited resources in terms of CPU and memory, thus, they participate in the system activities only when they have direct interest. Free-riders strategize their behaviour to benefit from the system at the lowest cost in terms of resources. These nodes are also called rational for their capability to maximize their participation in the system.

On the contrary, malicious behaviour is not driven by the intention to save resources but by the purpose to inject false data or to disrupt the system by impersonating nodes or by attacking nodes.

In real networks the fraction of malicious nodes compared to free-riders is low. This happens because nodes tend to acquire more resources if they are available in the system. For instance, in content distribution networks nodes are active in the system until they have downloaded the requested files. In the same network, the main activity of malicious nodes is to pollute the content available in the system by injecting bogus files or corrupted content. However, the impact of their behaviour causes more damages to the performance of the system and even to the user in case of viruses [86].

2.3.2 Adversarial model

In this section we discuss the malicious behaviour of one user or a group of nodes in an autonomic distributed system. The classification of the malicious entities, in accordance to [73], is done by considering the goal of the attack and the means to achieve it. The designer of the self-preservation mechanisms must evaluate the presence of the adversaries and implement an incentive mechanism to thwart possible attacks.

Inauthentic peers participate in the system by contributing to a transaction with a content that is different from the requested one. The effect of this attack depends on the type information they propagate and the use of this information. In a file sharing system, this attack can reduce the performance of the system in terms of content availability and download speed. In a context-aware application, false information can also mislead the contextualization of a service. However, if a node propagates a virus, this can cause higher damages to the peers that *benefit* the service.

Impersonation consists of *stealing* an identity and behaving like a legal identity in a transaction in order to reduce the level of trustworthiness of an honest node. In this type of attack, malicious nodes that pollute the system are not identified. Alternatively, the nodes can consume resources from the system without being detected.

Sybil attack consists of forging an identity and appearing in the network with temporary identifiers with the intention to disrupt the network or disseminate false information without being responsible for the action [40]. In general, the attacker enters the systems with multiple identities to simulate the control of multiple peers.

Denial of Service (DoS) attacks are typical of a client/server model where the attacker reserves or uses all resources available at the server to deny service to legitimate clients. This attack can seriously damage a self-organized system if its distributed version is used, for instance by using worms.

Repudiation consists in denying an actions. A node can refuse a transaction or to have sent feedbacks.

Whitewasher is a peer that behaves maliciously in the system. When other nodes refuse to serve this node because of its poor honest behaviour, it exits the system and enters with a new identifier to be labelled as new entrant [44]. This attack is possible when nodes are not bound to a fix identifier or when new entrants have an initial *credit* to spend in the system.

Bad Mouthing is an attack based on the recommendation a node sends. In this attack malicious nodes report false information in the system. This attack is effective when a distributed monitoring scheme is implemented and the functions of this scheme are based on the feedback the user sent after each transaction. A variant of this attack consists in sending false feedback for all peers except a small group. In this case, malicious nodes are not recognized immediately as liars and their attack has greater impact.

Traitors or *milking* agents are peers that have inconsistent behaviour. These nodes behave well for a period of time or in small transactions to increase their reputation inside the community and then start to behave maliciously to increase the profit of their action. This type of behaviour causes more problems in systems that give additional privileges to high reputed users. A typical example is given by the eBay system.

Collusion identifies malicious nodes that collude in order to cause more damages to a node or to the system compared to the single actions of individuals. The nodes joining a collusion can deny services to a node or they can behave in a way that the community identifies the victim as being malicious. This attack is particularly difficult to be detected as only the nodes part of the collusion are fully aware of the target.

Front peers work to increase their level of trustworthiness inside a community. When they are identified to be honest and cooperative nodes, these peers *recommend* other malicious peers to promote their activities. If these promoted nodes are part of the collusion, this attack can be also called *collusive bad mouthing*. This strategy is similar to the collusion attack, but the difference consists of the fact that front peers, or moles, do not target a specific victim.

Ballot Stuffing refers to the action of a group of nodes that report false transactions in order to increase their trustworthiness in the system.

2.4 Toward cooperation in autonomic communication systems

As discussed in Section 2.2, cooperation is a new *security* issue that must be tackled to ensure the survivability of autonomic communication systems. Nodes cannot be assumed to cooperate and fulfil their obligations toward the system goal. They have a selfish nature and there is no centralized authority that monitors the behaviour of the nodes and distributes the resources in accordance.

2.4.1 Theoretical approaches

Self-organized systems are also characterized by the presence of heterogeneous entities which might be under different administrative domains or authorities. In this context, cooperation cannot be assumed unless the system is managed or designed in such a way that individual selfish behaviour results in the system goal. If the *rules* of the system are designed to create the appropriate context for nodes to behave selfishly and, at the same time, to guarantee that the *social optimum* is achieved, cooperation is promoted inside the system. The design of the system's rules with this goal in mind is part of the *mechanism design* theory[76].

Part of mechanism design is to study how the preferences of the users can be aggregated efficiently. However, these preferences are private and nodes can misreport their preferences to modify the behaviour of the system. Moreover, the autonomic communication systems are in continuous evolution and adaptation and nodes can modify their behaviours to self-adapt to the new context of communication. For these reasons, *mechanism design* is not fully applicable in these systems. The distributed version of the theory is named *distributed algorithmic mechanism design* and it is still under definition. Definitions and details of the theory can be found in [42].

Game theory [53] is another powerful theory used to study the outcome of a system when nodes strategize their behaviour. To study the equilibrium of a system, an important assumption is to consider rational entities as their behaviour can be predicted. Rational nodes act to maximize their utility, which can be defined, in a simplified version, as the amount of resources they obtain compared to the resources they have spent to access them.

Game theory and mechanisms design are theoretical tools and simplifications must be made to study the complexity of a networked system. Thus, their primary use is to study the behaviour of the nodes and to have useful insights to understand how to reach an equilibrium point in the system. The existence of this equilibrium guarantees that the system has inherent self-preservation properties.

A considerable amount of research has focused on the exploitation of economic theories to model nodes' interactions. Most of this effort has been centered on the analysis of how selfish behaviour impacts the performances of peer to peer and ad-hoc networks [11, 46]. Previous work show that incentives schemes encourage cooperation in these networks and increase the benefit nodes obtain from their participation [45, 89].

In [21] an approach is proposed based on game theory to define and analyze a differential service incentive scheme to improve system's performances. Game theory is used to evaluate the amount of resources that a node must contribute to the system and the probability with which nodes' request will be served by others.

However, autonomic communication systems are also populated by malicious nodes and their action are difficult to be predicted as they do not follow any specific strategy. If cooperation is achieved in a fully rational environment, malicious nodes can easily get control of the system by attacking. Thus, starting from an evaluation of the system properties in a rational environment, countermeasures must be used to reduce the impact of malicious nodes. [43] revises the approach presented in [21] to study whitewashing and collusion attacks within a game theory framework.

Game theory is also used as tool to analyze the robustness of trust inference protocols in the presence of adversarial rational nodes [75], such as nodes joining in a collusion or whitewashing, defined in Section 2.3.2.

If we base the survivability of autonomous system on nodes' behaviour, the system might not properly function with respect to the designed goal: non-cooperative and malicious behaviour are predominant in the system. In unmanaged and distributed systems, incentive mechanisms must be used to eliminate malfunctioning component and to foster cooperation among selfish nodes.

2.4.2 Incentives mechanisms to foster cooperation

The result of uncooperative behaviour causes service degradation if not the complete system collapse. The area that has gained much attention to solve the problem of selfish and malicious nodes is peer-to-peer networks. The main reason consists of the growing number of users that use peer-to-peer systems for file sharing or for distributed storage.

A solution that has been implemented to guarantee the correct sharing of resources is the *tit-for-tat* strategy of BitTorrent [32]. This strategy consists of copying the strategy of the node at the previous interaction. In BitTorrent, the tit-for-tat mechanism is used by a node to decide whether to upload a portion of a file to the requesting node or not. The cooperation is measured as the amount of downloaded information from the requesting peer decreased by the uploaded bits. This scheme has been proven to be effective and to speed up the mean download time of a file [69].

However, in BitTorrent malicious nodes are not rated for their actions. Nodes can upload fake portion of file without being punished for their actions. Nodes have incentives to cooperate for the distribution of the file they are interested in, but they are not stimulated to cooperate for other resources. Moreover, the contribution of the nodes in providing different resources (i.e. participate in multiple torrents) is not considered. Preliminary work has shown that the implementation of an inter-resources collaboration mechanisms improves the performances of BitTorrent [57].

In the literature, monetary or credit-exchange schemes have been proposed to foster cooperation in peer-to-peer [56] and ad hoc [22] networks. In these schemes nodes receive a payment every time they participate actively in the system operations, e.g., serving a file to a node in a file sharing setting or by forwarding packets in an ad-hoc network. Micropayments rely entirely on cryptographic primitives and use virtual currency to remunerate good actions [95]. The access to resources is granted upon payment. This solution is rather effective if we can assume tamper-proof hardware to safely store credits or a trusted centralized entity that mitigates disputes in the system and accounts for credits. However, a centralized or accounting infrastructure is not applicable in a distributed system and the operations required by a monetary scheme might be cumbersome if nodes have limited connectivity or limited computational capabilities [61].

Another class of solutions is based on service differentiation: nodes that contribute more will get better services [58]. This solution anticipates the definition of reputation management schemes. Reputation is defined to give an estimation of the expected cooperation level of the user or nodes. In these schemes nodes' behaviour is monitored by a fraction of the nodes in the system and evaluated to construct the reputation value. Access to services and the quality of the service are controlled by the reputation of the requesting node.

Chapter 3

Building blocks for a reputation management system

3.1 Overview

This chapter discusses and defines the architecture and the framework of trust and reputation management systems. It presents the definition of the metrics to evaluate reputation management systems and it discusses security issues in reputation management systems.

Trust and reputation management systems are not intended to provide a security framework that protects the networks from traditional attacks. But they are proposed to give nodes that sense of trusted environment that is common in real human interactions, like social relationships. For this reason, we present trust and reputation schemes with the aim to provide *soft* security to nodes participating in a distributed system. Trust is a complementary mechanism to traditional cryptographic algorithms that preserve the system secure state from external and internal threats.

This chapter is not intended to provide algorithms and implementation details of the trust and reputation management systems. Instead, we discuss the basic components and available architectural solutions to design reputation schemes and we critically assess the suitability of existing techniques in autonomic communication systems.

3.2 Definition of entities and terms

This section defines the terms we use for the description of the general properties of a reputation management system and the entities that are part of the architecture. This section also defines the terms we use in the following chapters to study the application of reputation management schemes to an autonomic system.

Peer is the entity that participates in an autonomic system. It provides services and issues requests for services to other peers. In general, the peer acts on behalf of a user and it is identified in the system as a node. It has an identifier that is exposed for communication purposes, routing of messages or to be addressed by other peers. Multiple identifiers can be associated to a peer, as used in Sybil attacks [40].

Identity is a sequence of bits or a fully qualified name that identifies a node in the system. This identity can have local or broader scope and the choice is made by the system designer. For the reputation management system, the identity must be unique to identify a node and to reward or punish the node for its actions. The generation of the identity can be made from physical or networking address or the identity can be bound to a certificate.

Pseudonym is an artificial name used in transactions to avoid disclosure of the real identity. Peers can use pseudonyms to maintain anonymity or to avoid the linkage of their transactions. If the use of pseudonyms is needed to give entities control of their identity, it can cause troubles in the management of reputation schemes. The generation of multiple pseudonyms must be controlled to avoid whitewashing attacks [44, 47], defined in Section 2.3.2.

Cryptographic mechanisms must be used to generate pseudonyms in an efficient and controlled manner.

Malicious nodes are entities that try to attack the system with the intention to disrupt some parts or the whole network. They have a *byzantine behaviour* and they can inject false data, launch DoS attacks, not forward request or misreport. They [10]. Their behaviour is unpredictable and they do not follow any strategy.

Free-riders exploits the system for their own good. They try to maximize their utility and to contribute with minimum resources to the system.

Centralized authority is the trusted third party in the system. It has control of the transactions and maintains audit information of the users' activities. This is the single point of failure in a centralized system, which cannot function properly if the authority is unavailable. For this reason, DoS attacks often target the authority. Reputation systems, like eBay, use the centralized authority to rate nodes and to solve disputes. Distributed systems do not implement a centralized authority as this solution has two major drawbacks in terms of scalability and availability.

Structure and unstructured systems are two different types of overlay networks. Unstructured systems are characterized by nodes who connect randomly to other nodes in the system; ad hoc networks are examples of unstructured systems. Structured topologies use a protocol to organize node and assign protocol specific identifiers to nodes [90, 80, 85]. The network is organized in a Distributed Hash Table (DHT) to address nodes and route messages. Hybrid approaches exist and they combine different structured networks through an unstructured topology composed by supernodes.

Transaction is the interaction between two peers. In a transaction the role of the peers is to provide or request a service, e.g. file, CPU usage, packet forwarding. Due to networking failures, request might not reach the destination or the transfer of the service can be interrupted if the connection breaks down. Nodes consider unanswered or incomplete requests as null transactions or poor quality service in the latter case.

Cooperation is the action of a node that well behaves in a transaction. The term cooperation is largely used in the framework of theoretical game theory and it indicates the willingness of a node to share resources.

Defection is the opposite action of cooperation. Defection is the result of the uncooperative behaviour of the node. This behaviour is shown by denying access to resources.

Adversary is the term used to identify agents that act to harm the system or other entities. In this thesis we call adversary both malicious and selfish users if not specified. The reputation scheme must detect the presence of adversary nodes and exclude them from the system.

Reputation measures the trustworthiness of a peer in a system. It is the global system-wide view of a node or what is believed about this node. In short, reputation is the collective measure of trustworthiness based on the judgement of a community. It is quantified and it is calculated by considering the action of a node in the view of a community of users.

Trust is a relationship of reliance and decision in social science [62]. A trusted party proves to benefit the belief of other peers to fulfill its obligation or to return in future the *credit* received. The definition of trust might include also the concept of risk, when the value of the outcome of a transaction is high and there exists the probability of failure. The concept of trust is stronger than reputation as a node, that trusts, risks in person. Thus, it is subjective and it is formed based on personal experience or on the reputation that the node has within a community.

Opinion is the judgment that a node forms after a transaction on the quality of service received by the counter part. It is personal and the scope is limited to a single interaction. An opinion forms the so called private or first hand information resulting from own experience.

Confidence is the opinion a node has on the judging capabilities of a reporting node.

Feedback is the information reported to the virtual community after a transaction. Feedbacks are aggregated and contribute to the evaluation of the reputation value of a node.

3.3 The concept of reputation management

The definition of reputation management systems comes from the need to create a communication framework in online communities similar to traditional social relationships. The real application of these systems has many facets, as it is based on the context of communication and the underlying system definition.

The way reputation management systems must be deployed or how their outcome should impact the decision of the nodes is still not clear and depends on many factors, such as the application context, type of nodes and risk of the communication. Thus, to evaluate the effectiveness of reputation management systems it is important to assess how these systems improve the quality of online interactions and how virtual nodes value their use.

However, the use of reputation management systems is conditional to three properties [82]. Nodes must last for long in the system to count for future interactions. If the time that nodes remain online is short, they only look for the immediate outcome of the transaction. The second property consists of the willingness of nodes to distribute the feedbacks after the transaction and to consider them for their decisions. Finally, feedbacks and reputation should be useful for

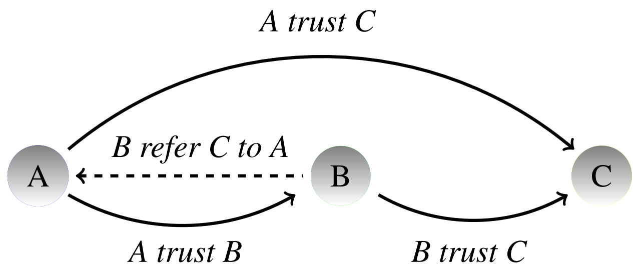


Figure 3.1: Trust transitivity as basic concept of reputation management systems. Node *B* is the connection point between node A and C

the community. To these properties we can add anonymity, as the peers' reputation must be preserved, and minimal overhead in terms of computation, storage and communication messages.

Fig. 3.1 shows the basic principle on which reputation management systems are based. For instance, the trust that *A* places on *C* derives from the recommendation that *B* issues, and on the confidence that *A* has on the capability of *B* to recommend other nodes.

Following the definition of the concept of reputation management, reputation schemes are also defined social control mechanisms as they secure the system from social behaviour attacks. In the framework of communication and computer security, reputation schemes are called *soft security* mechanisms to distinguish them from traditional security mechanisms like authentication, confidentiality and integrity. Accordingly, soft security mechanisms provide a type of trust that is called *provision trust* [62]. Provision trust gives the expected quality of service and the reliability of a node in a transaction. It differs from the *hard security* trust, as it does not guarantee that the system is protected from attacks.

In the following sections, the concepts of trust and reputation are further elaborated to define the basic framework for a reputation management system based on the *transitivity* principle depicted in Fig. 3.1.

Table 3.1: Architectural generalization of the reputation mechanism

	Centralized Systems	**Distributed Systems**
Collection	Send report to Trusted Third Entity	Send report to aggregation locations
Aggregation	Centralized computation	Distributed or local computation
Dissemination	Ask reputation value to Trusted Third Entity	Request reputation value of transacting node from aggregation locations

3.4 Breakdown of reputation management systems

Reputation management systems monitor the behaviour of their member nodes during transactions with other nodes and assist member nodes in selecting interacting nodes. To provide this service, the reputation system protocol must collect, aggregate and disseminate information concerning the nodes' behaviour. The collection and dissemination of the information depend on the system designer choices and they must consider the application and the underlying networking properties. Aggregation is performed to compute the trust value of a node, and the way information is combined is specific for the reputation scheme.

Hence a reputation management system performs three distinct functions: 1) collection, 2) aggregation and 3) dissemination of trust information. The first and the third functions require that nodes communicate whereas the second function, that of aggregating reputation information, may be performed locally at a single node or at multiple nodes depending on the reputation system architecture (shown in Table 3.1).

In this section we discuss the impact of the network topology and solutions to collect information from sources and to disseminate reputation values in the system. As aggregation functions are specific to a reputation management scheme, they will be briefly outlined and classified based on their general features.

3.4.1 Impact of the underlying network and topology

The underlying topology and the network properties of the communication have a great impact on the definition of a suitable architecture for reputation management systems. Two broader classes of approaches exist: centralized and decentralized communities. If the community is built on top of a centralized overlay network, the reputation system can rely on a trusted third party to collect and compute reputation values. The eBay feedback system is the most known reputation scheme that implements a centralized approach [81] and Slashdot [68] is an example of distributed moderation system that stores feedbacks in a centralized server.

eBay and Slashdot are web-based applications deployed on a client-server model. eBay is an acution site built in a centralized fashion as sellers and buyers negotiate the price by posting bids on the web. Only the final transfer of the good and the payment are done offline without the intervention of the server. This system automatically implies the possibility to exploit centralized properties to aggregate reports from the nodes and to publish the computed reputation value on the web.

Distributed systems do not assume the presence of a centralized entity and the load of the server must be distributed to the community. We can distinguish two topologies: unstructured and structured.

In unstructured systems, nodes have limited knowledge of the network and they interact only with neighbours or other entities they have met in the past. The reputation management scheme must follow the same structure of communication imposed by the protocol. For instance, P2PRep [33] implements a reputation management system for Gnutella, where each peer tracks the behaviour of the nodes in its transactions and share its view with other nodes by using a distributed polling algorithm similar to the one used to request resources.

Another example of unstructured networks is an ad hoc network where nodes are connected to their neighbours and have only local view of the topology network. As for Gnutella, reputation management systems rely on direct observation of the nodes and indirect opinions received by other entities [19, 74].

Structured overlays are organized topologies where nodes are *virtually* connected in accordance to the protocol. In these settings, reputation management schemes exploit the structure of the system to elect designated agents. These agents are responsible to store the reputation values of other peers and to aggregate feedbacks in some cases. These systems include P-Grid [8, 7], ROCQ [49], and PeerTrust [94].

A hybrid approach can also be implemented. A centralized authority can be used to manage the system initially or to provide assistance to new entrants and a distributed approach can be used to manage reputation. TrustMe [87] relies on a centralized server to bootstrap the system in order to implement anonymity and load balance schemes for storing reputation values.

Network properties must be considered in the definition of the reputation management architecture. In a mobile scenario, reputation schemes must deal with frequent disconnections and high churn rate, i.e. nodes that leave and join the system. In this scenario, we need to determine the extent to which it is possible to perform reputation management when the interactions are very short-lived. When the churn rate is very high, the design of a reputation management system must deal with high overhead to exchange messages to keep the reputation view consistent inside the system. Thus, before setting up a reputation management system in an ad hoc system, the overhead and the communication delays must be considered to quantify the possible benefit of reputation.

Structured peer-to-peer systems are also sensible to high churn rates as they need to reconstruct the topology [83]. In this case, multiple designated agents can be chosen to ensure resiliency and redundancy of the information.

3.4.2 Reputation aggregation locations

In this section we summarize possible aggregation and storage location for the reputation value. Depending on the system architecture aggregation of the collected feedbacks can be performed at a number of places in order to form the reputation value. A number of aggregation locations have been discussed in the literature [73].

- *Transacting Node*: Each node stores local opinions after each transaction. Only this information is aggregated to form a subjective view of the trustworthiness of the peers. Nodes only rely on direct evidence, i.e.first-hand experience, and do not share experiences with other nodes. Thus, the collection and dissemination functions do not need to be implemented.

- *All Nodes*: Nodes forms local opinions and share this information with all nodes that constitute the system. This solution is costly in terms of number of messages, as they increase quadratically with the number of nodes. Thus, the applicability is limited to small networks and when transactions are rare. This method is highly resilience to churn as nodes aggregate local opinions and indirect opinions locally. Nodes can weigh indirect opinions to reduce the impact of false information. Opinions from known peers or reputed peers can contribute more to the evaluation of trust.

- *Central Database*: This method relies on a central database to collect, aggregate and disseminate reputation values, like in eBay.

- *One-hop Neighbours*: Nodes share local opinions with one-hop neighbours. This scheme is suitable for ad hoc networks as it does not introduce huge extra-signalling to the system. A node may try to connect to multiple reputed neighbours to gather more information. The collected feedbacks might be weighted with a credibility factor.

- *Multi-hop Neighbours*: This is an extension of the one-hop neighbour as nodes share their first-hand experiences and the indirect information retrieved from other nodes they have contacted before.

- *Designated Agents*: Specific nodes are deputed to store and aggregate reputation values for other peers. These agents are named designated or score managers and an algorithm is used to select them. A simple mechanism to select a score manager in DHT is to hash the identifier of the node and assign the score manager role to the node responsible for the portion of the table identified by the result of the hash operation. This mechanism ensures that designated agents are randomly chosen from the entire node population and anonymity can be guaranteed as score managers do not need to know the identifier of the peer.

  Multiple designated agents can be used to store the reputation of one node. This number may depend on the churn rate and the fraction of misreporting agents in the system. Before a transaction, nodes contact the designated agents to retrieve the reputation information of the correspondent party and send them the opinion after the transaction. A credibility value can be used to weigh feedbacks and reputation values collected from nodes and score managers respectively.

3.4.3 Feedback collection

This section deals with possible approaches to collect information that pertains to the behaviour of a user in all of its previous interactions. The gathered information represents the input to the reputation aggregation function and can be collected in a proactive, reactive or hybrid way.

If the information is collected reactively, a node only sends its feedback when it is requested to do so, as shown in Fig. 3.2. This requires the reputation gathering service to periodically poll each node and ask them what feedback they have

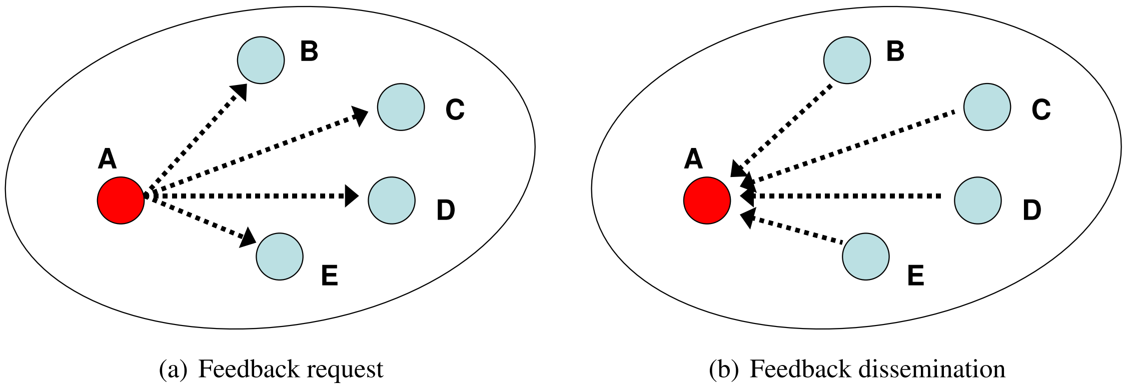


Figure 3.2: Reactive feedback collection: Node A – designated agent – periodically requests feedbacks from other nodes (a). Upon request, other nodes respond to A that is responsble for the collection and aggregation of feedbacks (b).

on nodes they have interacted with, after the previous round of feedback collection. The effectiveness of this approach is based on the frequency the designated agents query the other nodes. If the frequency is too high, nodes might have few interactions to report and the communication overhead introduced by the reputation management system is not compensated by a more accurate measure of the reputation value. On the contrary, if the frequency is too low, reputation values can be outdated. This approach has the advantage of not triggering any report actions after each transaction. The cost of this benefit is the delay in detecting nodes' misbehaviour if the information is not immediately reported.

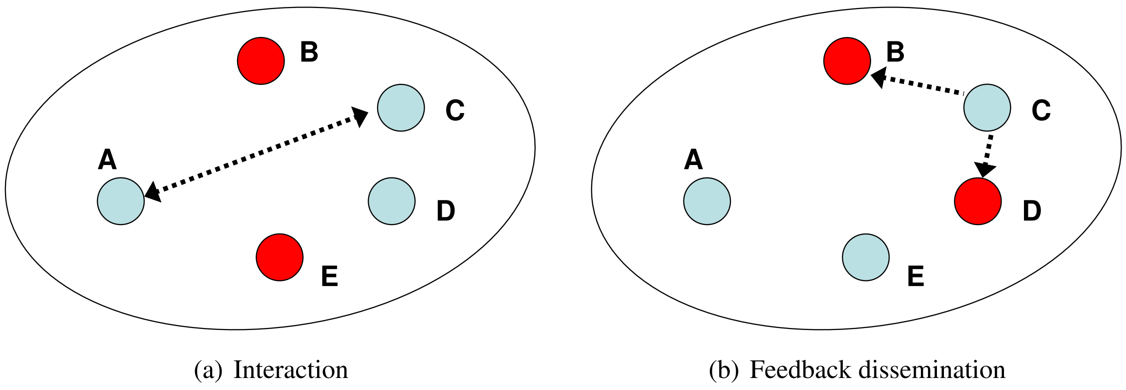


Figure 3.3: Proactive feedback collection: Node A and Node C interact (a). Node C sends feedback to designated agents of Node A (b) after the transaction.

If reputation information gathering is done proactively (see Fig. 3.3), a node sends the feedback on its transaction partners after every transaction. This approach enables fast detection of nodes' misbehaviour but may incur additional network overhead.

The hybrid approach consists of implementing a reactive and proactive scheme. A node periodically polls other nodes to gather information. However, if a node considers the importance of the feedback time-critical it may decide to disseminate this information immediately after a transaction.

3.4.4 Reputation aggregation functions

The reputation value of a node is calculated by aggregating the information pertaining to the history of the node itself. We must distinguish between two types of information: *private observation* and *public observation*. The former refers to direct experience of first-hand information and the latter to information that is publicly available. The aggregation function can use either public or private information or a combination of the two.

Several functions can be used to aggregate reputation values starting from a simple average of the feedbacks to a majority rule approach. In the latter, if a majority of nodes send positive feedbacks, then the node is assumed to be trustworthy. Other functions count first for positive and negative ratings separately and, then, define the score as the difference between the two computed values. eBay uses a similar mechanism to compute reputation [81].

More complex mechanisms that count for positive and negative ratings are Bayesian systems [62, 19, 20]. The beta probability density function is used to determine the expected probability θ for a node to misbehave, as shown in eq. 3.1. It is based on prior experience that is constantly updated by recomputing α and β as follows: $\alpha = p + 1$ and $\beta = n + 1$, where p and n are the number of positive and negative feedbacks respectively.

$$Beta(\theta, \alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1}(1-\theta)^{\beta-1} \tag{3.1}$$

where $\alpha, \beta > 0, 0 \leq \theta \leq 1$ and Γ is the Gamma function.

Other aggregation functions weigh each feedback F_ij issued by node i for the node j with other factors, e.g., the credibility of the reporter node C_xi estimated by node x, as defined in eq. 3.2. Other values can be used to weigh reputation, like a *quality* value that scores the importance of the transaction that nodes are reporting.

$$R_{xj} = \frac{\sum_i F_{ij}^{avg} \cdot C_{xi}}{\sum_i C_{xi}} \tag{3.2}$$

Using a quality value protects the system from *milking* agents that build up their reputation by behaving honestly in many minor unimportant transactions and, then, behave dishonestly in a few large transactions. If all transactions have the same weight, an agent can successfully milk the system by choosing a few large transactions to behave dishonestly in. The concept of transaction quality has been discussed in literature [94, 50].

The feedbacks can also be aged to assign higher weights to more recent experiences. This ensures that stale reputation information is aged appropriately and lower weight is given as reputation becomes older. The weight is function of the time when the feedback is created. For instance, if the information is t time units old, then, the weight of that information is computed as $e^{-\gamma t}$ where γ is the aging constant that denotes the rate at which information ages. Eq. 3.3 shows an example of the computation of the reputation value for node j made by node x, where i counts for the number of feedbacks to aggregate.

$$R_{xj} = \frac{\sum_i F_i e^{-\gamma t_i}}{\sum_i e^{-\gamma t_i}} \tag{3.3}$$

Other aggregating functions are based on a trust graph built from direct interactions between entities [94, 36]. Nodes are the vertices of the graph and the edges are the direct interactions. The weight w_e on the edge e rates the transactions. The reputation of a node j is calculated at node x by considering all the available paths from x to j and it can be expressed as shown in eq. 3.4.

$$R_{xj} = \sum_{e \in incoming(j)} w_e \cdot \frac{R_u j}{\sum_{f \in \text{incoming}(j)} R_u} \tag{3.4}$$

where R_u is the trustworthiness of the nodes that have interacted with node j and f gives the transactions that node u had with nodes connected to x. An algorithm can use this aggregation function recursively to determine the reputation values of all nodes in the system.

The choice of the best aggregation function and its implementation depend on the application context and on the desired output. The aggregated reputation value must be presented in a way that is useful to the consumers of this information. The aggregation service may output a binary value (trusted or not trusted), values on a discrete scale (say from 1 to 5 or $[-1, 0, 1]$) or on a continuous scale (0 to 1). A value of 0 means a node is not trustworthy and a value of 1 means a node is completely trustworthy.

3.4.5 Reputation dissemination

Once the reputation information has been aggregated to form a reputation value, it needs to be disseminated to other nodes. Similar to reputation information collection, reputation dissemination can happen both proactively and reactively. In proactive dissemination, reputation values for all nodes are pushed out to all nodes in the network at intervals. No dissemination is necessary if all nodes compute the trust values of all other nodes as in EigenTrust [63].

In reactive dissemination, a node has to explicitly request the reputation value of another node which it receives in response. In the case of multiple designated agents, a node receives several evaluations on the trustwhortiness of other nodes. These responses must be further aggregated to determine whether it will interact with the nodes in question. As for the case of the aggregation of feedbacks, a node can take the average as final value or a simple majority rule can be used to decide if a node is trustworthy. Responses can also be weighted with the subjective confidence value in the aggregating agents, similarly as defined in eq. 3.2 for reputation values.

Designated agents can disseminate other information, such as the number of transactions which the trust value is based on or the importance of those transactions. This can be useful to distinguish trust values that are based on very small number of transactions or on very minor transactions as trust values, thus, computed may not be an adequate indicator of the trustworthiness of a node. Moreover, the number of transaction is an important parameter to distinguish new entrants from malicious nodes.

To use the information gathered from designated agents, i.e., reputation values that we refer to public available information, a node can further combine reputation with its own personal experience. If we define the first hand information for node j computed at node x as opinion Oxj and the reports received from designated agents d, in general $d > 1$, Rdj can be combined as shown in eq. 3.5.

$$T_{xj} = (1 - w_p)O^{avg} + w_p \frac{\sum_d R_{dj}^{avg} \cdot C_{xd}}{\sum_d C_{xd}} \tag{3.5}$$

where T_{xj} is the trust node x calculates and w_p is the parameter to average the public and private information. However, when a node is interested in receiving a service it may be less interested in the trustworthiness of a particular node. Then, designated agents can disseminate a list of nodes providing the requested service that have a trust value above a given threshold. In this case, the requesting node can then contact these nodes with a request for service. If the first node cannot be contacted or refuses to provide the service, the requesting node can go down the list till it finds a node that will provide the service.

3.5 Metrics to evaluate reputation mechanisms

The performance of reputation management systems can be evaluated by considering several factors, such as communication and computational overhead or the percentage of correct decisions. In this section, we enumerate and discuss the most common metrics which are not specific to a particular application or implementation of reputation management schemes.

3.5.1 Success rate

The main objective of a reputation management system is to improve the performance of the system or the quality of service the nodes perceive. The first metric is the number of correct decisions, that nodes made, as a proportion of the total number of decisions made (see 3.6).

$$Success\ rate = \frac{\#Tr_{good} + \#Av_{malicious}}{Total\ \#\ of\ transactions} \tag{3.6}$$

where $\#Tr_{good}$ is the number of interactions with good peers that go ahead and $\#Av_{malicious}$ the number of avoided interactions with malicious peers.

A reputation system must discourage the interactions with malicious nodes. The decision is based on the level of trust of the interacting peers. Thus, if the reputation scheme predicts correctly the behaviour of the nodes, good nodes should have a high reputation value compared to malicious nodes. Free riders must not be able to receive services without offering any contribution and malfunctioning components must not have the possibility to disrupt good nodes or provide corrupted or dangerous services.

Thus, this metric evaluates if the reputation management system is able to detect untrustworthy behavior and if the system function as the collective desires.

3.5.2 Detection of misbehaviour

The detection of malicious nodes must be fast to reduce the risks of an attack and to avoid degradation of the system performance. Time and accuracy give an important estimation of the efficiency of a reputation management system. The reputation values must converge to the real expected behaviour of nodes within few interactions. In highly dynamic scenarios or in application, where the number of transaction is low, the convergence of the reputation management system is more critical.

Moreover, the estimation must be correct to avoid legitimate nodes to be identified as malicious or viceversa. A reliable reputation scheme must have few *false positives* in the shortest possible time. The good balance between time and accuracy depends on the specific application.

As we have defined in Section 2.3.2, there are several malicious behaviours or attacks that must be thwarted. Different metrics must be used to evaluate the performance of the reputation management system, one per type of attacks. Then, a global score must be determined to judge the goodness of the reputation scheme. The aggregation of the scores must depend on the type of the application, which the reputation scheme has been designed for, and the relevance of each security requirement and threat within the application context.

3.5.3 Communication overhead

The management of reputation requires the exchange of feedbacks among nodes in the system to estimate reputation values. The amount of load on the network components must be minimized to not overwhelm the normal functionality of the applications. The goal of reputation schemes is to manage entities in a self-organized fashion so that the perceived quality of the service increases. This extra signalling must be considered to judge the effectiveness of reputation management systems.

The evaluation of communication overhead must take into account the underlying topology of the network and the communication framework. In an ad hoc network formed by battery powered devices, flooding *reputation* messages might bring the network to a *congested* and non operational state. At the same time, few feedbacks might not be sufficient to define an accurate value for the reputation. The trade-offs between reliability and communication overhead is crucial in designing efficient reputation management schemes that do not limit the scope of the application.

The concept of extra-signalling is related to the property of scalability. Reputation management systems must also be able to work when the number of nodes might increase fast. In this setting, distributed management schemes are preferred to a centralized architecture as the load of managing reputation is distributed among the nodes. The property of scalability might not be required in specific applications, such as small communities of users.

3.5.4 Computational overhead

The load of the computation of reputation values is on the agents deputed to aggregate feedbacks, as discussed in Section 3.4.4. Every peer must also compute the trust value based on first hand experience and reputation values obtained from aggregation agents, as shown eq. 3.5. The number of operations required to process the messages is defined by the aggregation function and it is dependent on the number of messages and values that the agents must process.

In general, the highest computational cost is given by cryptographic operations used to ensure integrity, authentication and confidentiality of the message. The need for cryptographic algorithms is context specific and might be relaxed if the application is not critical or the devices are not capable of complex functions. However, most reputation management systems [49] require digital signatures combined with hash functions to ensure source authenticity and integrity of the information. The management of DHT tables to assign score managers is also based on the use of hash functions. But, the burden of processing cryptographic data might be cumbersome in particular when public cryptography is used. Thus, computational cost for messages is an important metric to evaluate the efficiency of reputation management systems in time-critical environments, such as short and frequent interactions.

3.5.5 Storage

The great availability of storage in modern devices does not impose high constraint for storing reputation values and feedbacks. However, the use of reputation management schemes in mobile environments populated by tiny devices might require particular attention to consider the amount of information that nodes can store.

In a large system, storing the reputation values of all nodes might be impracticable if nodes have low storage capabilities. In some cases, aggregation functions might require to save the complete history of a node, which can be large if nodes last for long in the system. Mechanisms based on timestamps can serve the goal of reducing the amount of stored data by discarding the oldest information to make space for new information.

However, reputation feedbacks can also be enriched with other information such as credibility, quality and the number of transactions. This information increases the cost of storage. In a centralized system, the data are saved in the central node and the cost of storage is linearly dependant on the number of nodes. In a decentralized system, the information can be replicated to achieve resiliency in case of node failures or to avoid attacks on second order reputation. In this last case, the evaluation of the performance of reputation mechanisms must consider the available resources at the nodes and the total space that must be reserved for feedbacks to make the reputation scheme function correctly.

3.6 Existing reputation management systems

In this section we discuss existing reputation management schemes for self-organizing systems. We present separately mechanisms suitable for ad hoc networks and for peer to peer networks as the context of the application is different. The ROCQ [50, 49] reputation management scheme is extensively discussed as we extend this model in the following chapters to evaluate the application of reputation mechanisms in different contexts.

3.6.1 Ad hoc networks

In [19, 20], the authors propose a reputation mechanism for mobile ad hoc networks based on Bayesian estimation. Each node monitors the behaviour of its neighbours and calculates trust values, similarly to eq. 3.1. This information is shared with other nodes which merge second hand information only if these values are close to the value estimated by the node or if the sources are trustworthy.

The authors propose a modified version of the Bayesian classification to consider a decay over time of the reputation values. In this way a node cannot capitalize from a good reputation earned in the past and malicious nodes can redeem themselves by behaving honestly in the system.

In [74], the authors propose a reputation scheme based on the watchdog mechanism. Each node monitors the behaviour of its neighbours with respect to the role of requester and provider of a function. The expected result is compared with the observed execution of the function to derive conclusions on the behaviour of the node. The scheme uses also indirect reputation to determine the trustworthiness of a node and to decide about cooperation or isolation of the node.

3.6.2 Peer to peer networks

Considerable research has been performed on reputation systems in order to motivate peers to collaborate and to penalize peers who cheat in peer to peer networks. Initial efforts at trust management in electronic communities is based on centralized trust databases [34, 81, 97].

Aberer *et al.* [8] introduce a reputation scheme that uses a decentralized storage system, *P-Grid*, to save trust information. P-Grid [6] is a virtual distributed search tree, similar to distributed hash tables, where each peer is responsible for a part of the tree. Replica of the same object can be placed in different positions of the tree.

The reputation mechanisms is based on negative feedbacks, i.e. complaints, given by nodes when the service they receive is not as expected. Peers file complaints against other peers who they feel have behaved maliciously and the complaints are redundantly stored at other peers called agents. Complaints are used to make a probabilistic assessment of the behaviour of nodes in past transactions. Two algorithms are described to compute the trustworthiness: the first relies on a simple majority of the reporting agents' decisions and the second evaluates the trustworthiness of the reporting agents.

To assess the trustworhiness of node, a peer queries the agents that respond to request with the complaints for the node. The mechanism is based on the evidence that a peer lies in reporting information to cover its own malicious behaviour. Given this assumption, a malicious node fills a complaint when providing bad services to good nodes as well as good nodes fills complaints. Thus, it will be difficult to find out who behaves maliciously in a transaction. However, if a node keeps acting maliciously many complaints will be available for the node and it will be identified.

The agents can lie when they report complaints. A complaint based mechanism is also defined for reporting agents. Reports coming from untrustworthy agents are not taken in consideration. This scheme is one of the first approaches that model the ability of the nodes to make recommendations.

Kamvar *et al.* [63] propose EigenTrust, a decentralized reputation management system that provides anonymity, limits the benefit for newcomers and is robust to collusion attacks. The scheme is based on the property of transitive trust, as a peer weighs the trust ratings it receives from other peers by the trust it places on the reporting peers, resulting in a mechanism similar to the one defined by eq 3.4.

Peers store the subjective view of the trustworthiness of other peers in a local matrix. The trust values are normalized in the interval $[0, 1]$. These values are the weights a node uses to rate the reports from its neighbours to calculate the global trust values. The algorithm to determine the global trust values is distributed and the results correspond to the left principal eigenvector of the updated matrix. The same global trust values must be obtained by the peers in the system, but peers can still modify their own ratings. A DHT is used to store anonymously multiple copies of the same trust value for a node at several peers. A set of pre-trusted peers is assumed in the system to make the scheme also robust to collusion attacks.

PeerTrust [94] is another reputation management system that considers 5 factors to compute the trustworthiness of a node. These are: 1) the feedback obtained from a node, 2) the number of total transactions a peer has with other nodes, 3) the credibility of the node reporting the feedback, 4) the context of a transaction to differentiate between important and non-important transactions and 5) the community context factor to derive the characteristics and vulnerabilities of a community.

The feedbacks are the means to estimate the beahviour of a node and a feedback is interpreted as the amount of satisfaction the transacting peers receive. The credibility of the feedback source is used to asses the quality of the report and how this report should count for the trust value. Two metrics can be used to estimate the credibility. The first is based on the trustworthiness of the reporting nodes, similarly to the EigenTrust. The second considers the credibility as the similarity in amount of satisfaction of two peers over a common set of nodes they have interacted with. The transaction context factor is used to distinguish transactions based on the size, importance and freshness. More recent transactions should have a greater weight than small and old ones. The community context factor is propose to incentivize nodes to give feedbacks.

The metric to compute trust is a weighted average of the satisfaction a peer receives for each transactions. Credibility and transaction context are the weights for the trust function, and the number of transactions and the standard deviation of the feedbacks are used to compute a confidence value. The result is adjusted based on the community context factor.

3.7 The ROCQ model

Garg *et al.* [50, 49] recently propose the ROCQ scheme (Reputation, Opinion, Credibility and Quality), a reputation-based trust management system that uses opinions from past interactions to measure the trustworthiness of a peer. In particular the reputation of a node is calculated by considering the credibility of the reporting peers and the first-hand opinions of the users. ROCQ also introduces the quality of an opinion as a parameter to express the emphasis that a node poses on its feedback. This is useful when the transaction is less important or when the reporting peer is not sure of its opinion. The global reputation values are stored in a decentralized fashion and score managers are designated to compute and store reputation values anonymously.

3.7.1 Opinion

A peer forms an opinion about the amount of satisfaction it has derived from a transaction which it takes part in. The term O_{ij}^k refers to peer i's opinion about its k^{th} transaction with peer j and is normalized to take values in $[0, 1]$. A peer keeps a record of its own first-hand experiences in the form of averaged opinions. O_{ij}^{avg} is the average amount of satisfaction that peer i has received from peer j (eq. 3.7) in N_{ij} interactions.

$$O_{ij}^{avg} = \frac{\sum_{k=1}^{N_{ij}} O_{ij}^{k}}{N_{ij}} \tag{3.7}$$

3.7.2 Reputation

The reputation of a peer j is the result of the feedbacks' aggregation computed by the score managers. It represents the global system-wide view of the average amount of satisfaction a peer is likely to derive through an interaction with j. The reputation of a peer is also normalized so that it lies between 0 and 1.

$$R_{mj} = \frac{\sum_i O_{ij}^{avg} \cdot C_{mi} \cdot Q_{ij}}{\sum_i C_{mi} \cdot Q_{ij}} \tag{3.8}$$

Eq. 3.8 shows the aggregated reputation R_{mj} of peer j computed at the peer m. C_{mi} is the credibility of the reporting peer i according to peer m. O_{ij}^{avg} is the reported average opinion of peer j by peer i and Q_{ij} is the associated quality value sent by reporting peers. Thus, peer m weighs more ratings that are considered to be of a high quality and that come from peers that he believes to be more credible.

3.7.3 Credibility

In ROCQ, credibility ratings are based on first-hand experience only and, unlike opinions, they are not shared with other peers. Credibility ratings are normalized so that they lie between 0 and 1. If a peer gives wrong feedback about other peers its credibility rating is decreased and its subsequent reports have a reduced impact on the reputation of another peer. Similarly, if a peer's feedback is consistently good, i.e., in agreement with other reporting peers, its credibility rating goes up.

The credibility is computed upon the agreement of the reported opinions and the computed reputation value. When a peer reports an opinion to another peer for the first time, its credibility is set to 0.5. Thereafter, on the $k+1^{th}$ report to peer m, the credibility of peer i is computed as shown in eq. 3.9.

$$C_{mi}^{k+1} = \begin{cases} C_{mi}^{k} + \frac{(1-C_{mi}^{k}Q_{ij})}{2}(1-\frac{|R_{mj}-O_{ij}^{avg}|}{s_{mj}}), & \text{if } |R_{mj}-O_{ij}^{avg}| < s_{mj} \\ C_{mi}^{k} - \frac{C_{mi}^{k}Q_{ij}}{2}(1-\frac{s_{mj}}{|R_{mj}-O_{ij}^{avg}|}), & \text{if } |R_{mj}-O_{ij}^{avg}| \geq s_{mj} \end{cases} \tag{3.9}$$

C_{mi}^{k} is the credibility of peer i after k reports to peer m, O_{ij}^{avg} is the opinion being currently reported by peer i, Q_{ij} is the associated quality value, R_{mj} is the aggregated reputation value that peer m computed for j and s_{mj} is the standard deviation of all the reported opinions about peer j.

3.7.4 Quality

ROCQ allows a peer to determine the confidence of its feedback. Giving incorrect feedbacks can decrease the credibility of a peer. So, a peer can lower the quality value for opinions about which the opinion are not consistent or they are few. Quality is also normalized to lie between 0 and 1.

We assume that the expected amount of satisfaction during each transaction for peer j is a normally distributed random variable. Through interactions with j, peer i makes observations of this random variable resulting in samples. The sample mean and standard deviation are then O_{ij}^{avg} and s_{ij}.

The quality value of the opinion (Q_{ij}) is defined as the confidence level that the actual mean trust rating for a peer lies within the confidence interval. Further details on how quality values are computed can be found in [50].

3.8 Security considerations

This section discusses the security problems that arise in the context of reputation management systems. Reputation mechanisms cannot prevent all kinds of attacks that malicious entities can inflict on autonomous system as they are *soft mechanisms* that attempt to enforce good behavior in the network through incentives. Hence, attacks on the integrity of the underlying network must be thwarted with *hard security* mechanisms. We describe some of these attacks, discussed in Section 2.3.2, and solutions that have been presented in the literature to detect nodes that, for instance, mis-route messages or impersonate entities or launch sybil attacks to subvert the system.

3.8.1 False context information

Autonomous systems usually create an overlay network for communication between the constituents. These overlay networks require collaboration among users to forward messages and to update routing tables in a consistent manner. Malicious entities can disseminate false routing updates in the network and they can falsify information about the location or status of a resource so as to keep control on the resource.

However, failure in message delivery can be caused either by topology changes due to the dynamic nature of such networks or by a compromised node. As a result, it is not trivial to distinguish between benign and malign failures. Because the operability of such networks depends on the nodes' willingness to forward messages, a compromised node will affect more than just the interactions it is part of. Failure tests or redundant [27] and iterative routing [88] can be used to detect faults and to provide alternative routes around the failed node. But this results in a cost increase since multiple paths need to be stored and updated.

3.8.2 Repudiation and identities

Possible attacks on autonomous systems include denying the existence of data the node is responsible for and repudiating transactions, as discussed in Section 2.3.2. The first type of attack can be avoided by carefully replicating the object at different locations not under the control of the same node. This approach can be beneficial when the data stored can be tampered or mislabeled.

However, such a replication scheme will not be effective when a node can assume multiple identities. This attack, known as *sybil* attack [40] and presented in Section 2.3.2, can severely compromise an autonomous network, since malicious nodes can counterfeit identities with different reputation values and control different object in the system. This eliminates any benefit that could have been obtained from a reputation management scheme, from replicating data or from using alternative paths for routing.

In [40], Douceur suggests using a trusted identification authority that is in charge of establishing node identifiers. This solution solves the problem of sybil attack, but introduce some limitations in the system. The trusted identification authority must be a centralized entity that must ensure the release of only one identity or one valid certificate per node. Thus, to be effective lost and revocation of compromised or stolen identities must be implemented. This is a complex task to achieve in large distributed systems.

In [27] certified nodeIDs are proposed; this requires each entity to own a certificate (valid public/private keys) and binds the nodeID to a specific IP address. This solution has several drawbacks since node mobility or changes in the network can cause the node's IP address to change requiring re-certification or the assumption of a new identity. Furthermore, the solution does not work for nodes behind a NAT.

The presence of a certification authority allows a public key infrastructure to function, which in turn allows for encryption of messages and signing of feedback messages. This ensures the integrity of trust queries, replies and feedbacks in transit. Furthermore, a peer cannot repudiate a message once it has been sent and recipients are able to verify the identity of the sender and the authenticity of the message.

Other solutions to defend against sybil atttacks consist of resource tests, such as the use of cryptographic tools to reduce the rate at which nodes can generate a new identity. For instance, crypto puzzle can serve this goal by requiring new nodes to solve complex task before joining the system. These resource tests do not solve completely the problem *sybil*, but they discourage nodes from creating new identities as they are resources consuming. However, this approach is effective if there is a central authority that manages the bootstrap of the the nodes or a distributed authority, formed by pre-trusted peers, that helps nodes in the process of joining.

Chapter 4

Extra-signalling in reputation management schemes

4.1 Overview

Self-organized systems are composed by heterogeneous components that interact to disseminate and collect data in order to accomplish complex tasks, such as to create new services that evolve in the system and adapt to the context of the communication. Cooperation is required to achieve scalability of the system and maintain survivability of the system which evolves continuously. Components must follow their obligations to enhance content availability in the system.

Nodes might act selfishly in the sense that they do not share the data they own or they can be malicious by injecting false content. In this potentially non-cooperative environment, trust and reputation management schemes have the key role to sustain the stability of the system and reputation information to function as fitness criteria for service evolution and adaptation.

However, the presence of reputation information introduces an extra signaling induced by the operations to manage trust in the system and to disseminate and collect feedbacks from the nodes, as discussed in Chapter 3. While we can assume that the feedback collection is part of the process required for the loopback process control of autonomic communication systems, the overhead induced by the mechanism must be considered in order to evaluate the effectiveness of injecting reputation information, as presented in Section 3.5.

For instance, mobile networks are formed primarily by battery-powered components, thus, the number of messages/transactions must be reduced to a minimum to increase the survivability of the system. In peer to peer networks, malicious nodes can provide service of poor quality or corrupted content to reduce the performance of the application. Trust and reputation can be used to decide whether to interact with a node to increase network resilience and to reduce the number of malicious transactions. But this benefit of reducing attacks must be quantified by considering the extra signaling of reputation management schemes.

This chapter addresses the problem of communication overhead of reputation management systems defined in Section 3.5.3.

4.2 Networking of reputation systems

The definition of reputation management systems comes from the need to create a communication framework in online communities similar to traditional social relationships. The real application of these systems has many facets, as it is based on the context of communication and the underlying system definition. The way reputation management systems must be deployed or how their outcome should impact the decision of the nodes is still not clear and depends on many factors, such as the application context, type of nodes and risk of the communication. Thus, to evaluate the effectiveness of reputation management systems it is important to assess how these systems improve the quality of online interactions and what is the *cost* of their use.

As pointed in Section 3.4, a reputation management system is composed of three functions to 1) collect feedbacks after transactions, 2) aggregate them to form a useful measure of trustworthiness and 3) disseminate the reputation value of a particular node to requesting members.

In this chapter, we discuss architectural solutions and analyze the impact of the collection and dissemination schemes as the aggregation does not induce extra communication signaling in the system. The aggregation takes place at the designated agents or locally at each node in the system.

4.2.1 Networking topologies

There exist two main types of network architecture: centralized and decentralized. In this chapter we only analyze decentralized systems as we can assume that the communication overhead increases linearly with the number of nodes in centralized systems. Distributed systems do not assume the presence of a centralized entity and the load of the server must be distributed to the community. We can distinguish two topologies: unstructured and structured.

In unstructured systems, nodes have limited knowledge of the network and they interact only with neighbours or other entities they have met in the past. Structured overlay are organized topologies and nodes are *virtually* connected in accordance to the protocol. The reputation management scheme must follow the same structure of communication imposed by the protocol. In these settings, reputation management schemes exploit the structure of the system to elect designated agents. These agents are responsible to store the reputation values of other peers and to aggregate feedbacks in some cases [8, 49, 94].

Network properties must also be considered in the definition of the reputation management architecture. In a mobile scenario, disconnections are frequent and there is a high churn rate, i.e. nodes that leave and join the system. The interactions are very short-lived and it is important to determine wether reputation management can be performed. Moreover, when the churn rate is very high, the design of a reputation management system must deal with high overhead to exchange messages to keep the reputation view consistent inside the system. Thus, before setting up a reputation management system in an ad hoc system, the overhead and the communication delays must be considered to quantify the possible benefit of reputation.

Structured peer-to-peer systems are also sensible to high churn rates as they need to reconstruct the topology [83]. In this case, multiple designated agents can be chosen to ensure resiliency and redundancy of the information.

4.3 Information dissemination

The collection of feedbacks and the dissemination of reputation information can be based on proactive, reactive or hybrid approaches. The hybrid scheme is defined as a combination of the reactive and proactive approaches. The reputation management system does not require specific communication protocols to achieve its functionality. Communication depends on the network infrastructure available and overlay network on top of which the reputation management system is built, as defined in Section 3.4.1.

To reduce the overhead of messages, that are required by the reputation system to function properly, several mechanisms can be in place such as piggybacking the information to application messages or aggregate multiple feedbacks in one message. As defined in Sections 3.4.3 and 3.4.5, different strategies can be used to disseminate feedbacks and reputation values, and they depend on the scope of communication.

Feedbacks are formed after transactions locally and reputation values can be aggregated at several location, as defined in Section 3.4.2. To generalize the results derived in this study, we analyze a reputation management system where there exist designated agents to aggregate and store reputation values. Designated agents are special nodes that only monitor the activities of the members of a community, but they are part of the community themselves and they interact with other nodes to provide and receive services.

In our study, we also consider multiple replicas of the same reputation value to derive conclusions on the system performance in terms of communication overhead and the accuracy of the prediction of the behaviour of the nodes. Designated agents can be chosen by simply hashing the identifier of the node and looking at the DHT table to find the agent responsible for the resulting key in the table. Multiple agents can be defined by using different hash functions or by concatenating the identifiers with other information.

For the sake of completeness, we discuss the setting with designated agents, in particular when multiple designated agents are available to aggregate and store the reputation of a node.

4.3.1 Collection of feedbacks

In Section 3.4.3, we have shown that a proactive approach requires nodes to send feedbacks after each transaction. A transaction is defined as an interaction between two nodes and it depends on the application context. Few examples are the transfer of a file or part of it, the forwarding of a message and service provision. Thus, nodes must locate designated agents and send one message per agent. This approach enables fast detection of nodes' misbehaviour but may incur additional network overhead, as nodes must locate designated agents and send one message per agent.

If the information is collected reactively, a node only sends its feedback when it is requested to do so. The reputation collecting service periodically polls all nodes to gather new feedbacks of all their past transactions or aggregated values since the previous polling round, as defined for a reactive approach (see Sections 3.4.3 and 3.4.5) The effectiveness of this approach is based on the frequency the designated agents query other nodes. If the frequency is too high, nodes might have few interactions to report and the communication overhead introduced by the reputation management system is not compensated by a more accurate measure of the reputation value. On the contrary, if the frequency is too low, reputation values can be outdated.

In the reactive approach, the message polling must be initiated by designated agents. Otherwise, nodes can send messages containing feedbacks at regular intervals. The reactive approach can be defined with two different strategies for sending feedbacks: 1) upon receiving a request, each node broadcasts all available information to the network and the designated agents process the data for the nodes they are responsible for or 2) each node groups the feedbacks that are relevant to each designated agent and sends the messages separately.

The first strategy has the advantage of reducing the total number of messages in the system required to send feedbacks, but it has the disadvantage of asking every designated agent to process every single message even if it is of no interest for the node. If we consider that every node in the system can be a designated agent, the total computational cost is huge. The second strategy increases the total number of messages compared to the first one, but designated agents only receive the information, they have asked for.

From a reputation perspective, the hybrid approach requires designated agents to poll nodes in the system to gather opinions periodically and, for time-critical interactions, nodes can send feedbacks immediately after a transaction or they can request a reputation value of a node.

4.3.2 Dissemination of reputation values

Reputation values are aggregated and stored at designated agents, which can, then, disseminate reputation information for a specific peer to requesting nodes or they can push all the information they hold in the system. The most efficient method in terms of reaching all the nodes in the system is *flooding*, i.e., messages will be forwarded to all nodes in the community even if they are not interested in the content. In this case the overhead might be huge and the cost of the extra signalling may not be compensated for by the benefit of flooding. Better strategies consist of forwarding the messages by inserting in the system *K-copies*, with $K < N - 1$ where N is the number of nodes, of the same data to reduce the time for the dissemination and at the same time the number of messages.

These approaches enhance the robustness of the system in case of disconnected networks or in case the system is dynamic and the topology can change frequently. Flooding or *lighter* versions of flooding could be used in highly connected topologies as few messages are sufficient to reach all nodes in the system. On the contrary, in mobile networks, processing a great number of messages can consume the battery of the devices fast.

Multiple designated agents must exist in the system as it is important to avoid attacks of a malicious designated agent who can falsify reports. This choice guarantees resilience and robustness in the network. For instance in a mobility scenario it is not possible to assume that all designated agents are always present during the whole lifetime of the system. However, the use of multiple designated agents increases the cost of communication as the same information is disseminated by every agent.

4.3.3 General considerations

Nodes should also maintain a record of all transactions, they have participated in, as they form an opinion on the amount of satisfaction the node receives. Locally, nodes can cache temporary reputation values of all the nodes in the system or only for a subset of nodes with whom they have already interacted. This information is useful when interactions between the same nodes are frequent or the communication links are congested or frequently disconnect as for mobility scenarios.

Since storage capabilities of nodes are limited, out-of-date reputation values or transactions' feedbacks can be safely filtered and discarded with mechanisms based on timestamps. In this context, the timestamp, used to age reputation values or opinions, can function as indicator of the freshness of the data provided. This has a twofold meaning: 1) reduce the amount of information stored at designated agents and 2) age both the information of the reputation aggregation function and the computation of the trust value, as defined in Sections 3.4.5 and 3.4.4.

In the following sections, we assume the autonomous system is structured and organized by using a distributed hash table (DHT). This assumption is needed to define the overall cost of reputation management systems in specific topologies. In particular, we assume that the routing of messages is done by means of the DHT and not with a gossip-based mechanism which might be used in unstructured networks [98]. However, we do not deal in this chapter with the messages required to build and maintain the DHT.

4.4 Types of reputation

In reputation management systems, the burden of the communication is mainly on designated agents which have the function of collecting, aggregating and disseminating reputation values. The communication is based on packets or on the notion of messages. Messages should consist of a payload and metadata that carries the necessary information for a node to process the payload. To decide the type of information that forms a feedback, we refer to the ROCQ reputation mechanism, presented in Section 3.7.

In this section we discuss different types of *reputation* that can be associated to a node and, at the same time, provide a brief discussion on the use of reputation values. Nodes store up to three different classes of reputation information:

- **First-hand information on nodes (known also as *opinion*** (O_n) is the amount of satisfaction a node receives after a transaction. This information is useful for forming the feedbacks that they send to *designated agents* responsible for aggregating reputation values for the transaction partner.

- **Second-order information on nodes (known also as *credibility*** C_n) is used to weigh feedbacks from nodes or reputation values received from designated agents. Credibility is defined as a measure of the judging capability of the reporting node.

- **Reputation information on nodes** (R_n) is the global system view of the behaviour of a node. It is maintained locally by designated agents for nodes they are responsible for or in some case by the node in the system, as defined above.

4.4.1 Opinion

Each transaction between two entities results in the formation of an opinion on amount of satisfaction measured for the transaction. Then, this opinion is sent as feedback along with a timestamp that indicates when the transaction took place. The feedback also consists of the confidence the reporting node has on the opinion, i.e., the quality value of the opinion.

Feedbacks can be represented as a tuple (O, TS) where O is the feedback and TS is the timestamp. The size of the feedback depends on the number and type of fields. In total, one feedback can be encoded by using 28 bytes and it consists of:

- **Node identifier** Id is encoded on a fix number of bytes and it depends on the choice of the designer, if we use a MD5 hash function to determine the identifier, this can be represented with 16 bytes. This identifier is required not for forwarding decisions but to record correctly a transaction. The hash value of the identifier can be also used to provide anonymity for the node and avoid possible collusion attacks [8, 49, 63].

- **Timestamp** TS is used to age the information and it can be encoded by using 4 bytes. This number can be further reduced by defining different formats.

- **Opinion value** O is the subjective view of the node. This value is between 0 and 1. A value of 0 means a node is not trustworthy and a value of 1 means a node is completely trustworthy. Opinion values can be encoded on 4 bytes.

- **Quality of the opinion** $Q(O)$ is the *confidence* associated to the opinion value. This value might be used for reputation aggregation at designated agents as it quantifies the confidence a node has on the reported opinion, more details can be found in [49, 50]. This value can be encoded by using 4 bytes.

4.4.2 Reputation

Designated agents collect feedbacks encapsulated in messages either in reactive or proactive mode, as defined above. In reactive mode, designated agents polls the nodes to retrieve opinions or, also, the nodes send their opinions in response to such a request (thus, reactive) by a designated agent. In proactive mode, nodes report immediately their opinions without waiting for being asked for. The cost of this operation is proportional to two factors: 1) the number of lookups in the DHT, as we assume multiple designated agents, and 2) the cost of each lookup. The latter depends on the DHT scheme and it is represented by the number of messages to reach the designated agents, which can be assumed to be the order of $O(logN)$, as for Chord [90] and Pastry [85], where N is the number of nodes in the DHT.

Before each interaction, nodes ask the designated agents for the reputation value of the correspondent party. When providing reputation information, designated agents may misbehave and send false information. Thus, the second-order reputation is used to weigh responses, as discussed in Section 4.4. The reputation value can be further aggregated with the local opinion by using metrics similar to eq. 3.5, discussed in Section 3.4.5.

The payload of the message carrying the reputation value has a length of 28 bytes plus the node identifier and it consists of:

- **Node identifier** Id is encoded on 16 bytes as defined earlier.
- **Timestamp** TS is used to age the information and it can be encoded by using 4 bytes. This number can be further reduced by defining different format for the timestamp.
- **Reputation value** R is the reputation value aggregated at designated agents. This value is between 0 and 1. A value of 0 means a node is not trustworthy and a value of 1 means a node is completely trustworthy. Reputation values can be encoded on 4 bytes.
- **Quality of the reputation value** $Q(R)$ is the *confidence* associated to the reputation value. This value might be used for reputation aggregation as it weighs new reports into the reputation value R by considering the importance of a transactions. This value can be encoded, like the timestamp, by using 4 bytes.

The dissemination is done in a way similar to the collection. If a hybrid approach is implemented, trust values are pushed periodically to nodes and in case of need they can be retrieved on-demand from designated agents. This message contains the reputation value for each node the designated agent responsible for and each value is represented by the tuple (Id, R, Q(R) and TS).

The overhead depends on the frequency of the messages, thus, the rate must be chosen carefully to not overload the system with reputation messages. The rate must also consider the number of nodes that are present in the system. An adaptive scheme can be used for this purpose: if the number of nodes is high, the rate will be reduced and nodes will rely on a proactive approach if they require up-to-date reputation values.

4.4.3 Nodes credibility

Credibility, or second-order trust information, is computed by designated agents on all nodes reporting opinions and by nodes on the designated agents. Credibility is not shared with other nodes and is evaluated on a direct-experience basis only.

Initially, the credibility of all nodes is set at 0.5 by default and designated agents nodes can increase or decrease their credibility by reporting accurate or inaccurate information. The second-order trust information is computed locally and does not require input from other nodes. Thus, it does not introduce communication overhead. Eq. 3.9 shows the metric to compute the credibility factor for ROCQ.

Credibility is used to aggregate received trust information. For instance, multiple designated agents compute the reputation value for a node . A designated agent, misreporting reputation information, can be detected by comparing the received reputation scores from other designated agents. Its credibility is adjusted in accordance.

4.5 Implementation details and experimental results

In this section we evaluate the extra signalling introduced by the reactive and proactive approaches and we analyze the impact of different parameters on the performance of reputation management scheme. This analysis is required to understand the benefit of reputation management systems compared to the waste of resources due to the communication overhead.

In our experiments we use a simplified model of the ROCQ reputation scheme [49], discussed in Section 3.7. The parameters we use for the experiments are listed in Table 4.1. We run a number of initial transactions equal to the number of nodes to bootstrap the reputation management system. One transaction is counted per iteration and the two nodes are selected randomly

Table 4.1: Parameters' setting for overhead estimation

Network Topology	random
Number of iterations	45,000
Experiments run	5
Malicious Nodes	30%
Type of maliciousness	Transactions and Feedbacks
Collection of feedbacks	Proactive or Reactive
Dissemination of reputation values	Proactive or Reactive
Frequency of polling	every 2,500 or 5,000 iterations
Threshold for trustworthiness	0.5
Designated agents	5 (if otherwise specified)

from the population.

At each iteration corresponds a transaction, the two nodes compute the trust value of each other and if this value is above the deterministic threshold 0.5, the interaction takes place. The computed trust value is a function of the opinions, that a node forms after each transaction, and of the reputation value aggregated at the designated agents, as presented in eq. 3.5. The use of the opinions or the reputation value depends on the strategy the node selects.

Our model targets malicious nodes that misbehave when interacting with other peers, when reporting feedbacks to score managers or when reporting reputation values to nodes in the role of designated agents. Multiple score managers are used to mitigate the effect of malicious agents in the system. Reputation values of the nodes are aggregated at designated agents and they are computed by using the feedback received O_i, the credibility of the reporting node C_i and the confidence the reporting node as on the feedback $Q_i(O)$, i.e. the quality value in ROCQ.

The performance is evaluated in terms of the success rate, defined by eq. 3.6: the number of correct decisions made (i.e., interactions with good peers plus the number of avoided interactions with malicious peers) as proportion of the total number of decisions made. Only decisions made by good peers are counted. Moreover, we determine the number of messages exchanged due to the reputation management scheme and evaluate the impact of the number of messages on the success rate. Two approaches are considered:

Proactive: the reputation value of the interacting node is requested to the designated agents before each interaction and the feedback is sent after the interaction.

Reactive: feedbacks are collected at regular intervals and nodes send to designated agents only new reports since the last polling. Before sending the information, each node aggregates the reports that it must send to each designated agent. The reputation values of all nodes in the system are stored locally and they are updated when the designated agents disseminate new values. We simulate two frequencies for collection and dissemination: every $2,500$ iterations and $5,000$ iterations.

4.5.1 Overhead messages

Fig. 4.1 shows the number of overhead messages for the proactive and reactive schemes. We only consider the amount of messages the reputation management scheme requires for the collection and dissemination of information. As expected, the number of messages is highly dependant on the number of nodes for the reactive scheme, while it is independent for the proactive scheme as feedbacks and reputation values are requested for each transaction. For this approach, the amount of overhead depends on the number of designated agents, which is discussed later in Section 4.5.3.

We simulate two different strategies for the node to decide if carrying on a specific transaction with the other node: only the reputation of the other node or a combination of the reputation and the opinion, if previous transactions exist between the same two nodes. In the former, a node decides to go ahead in a

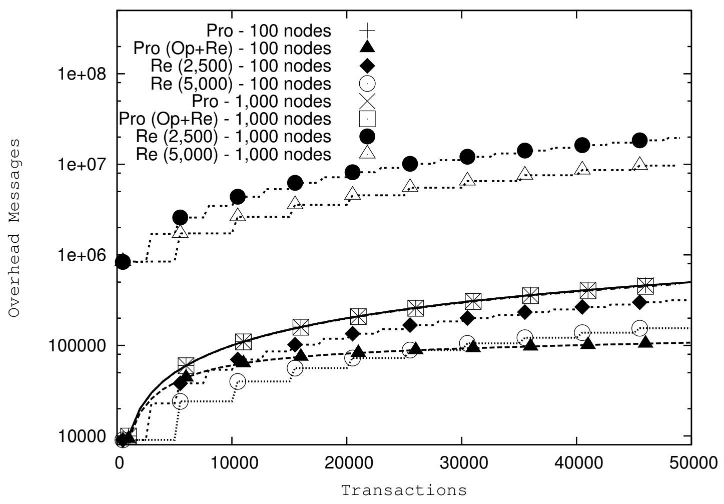


Figure 4.1: Overhead messages required by the reputation management scheme to function in a system composed of 100 or $1,000$ nodes when the Proactive (*Pro*) approach or the Reactive (*Re* - polling every $2,500$ or $5,000$ transactions) approach is used. If not specified otherwise, only reputation is used as the metric to evaluate the trustworthiness of a node for the forthcoming transaction. Only for the proactive approach *Op+Re* is used as a metric, i.e., reputation is used if there are less than 5 direct interactions.

transaction if the correspondent party has a reputation value above the threshold 0.5. In the latter case, the reputation value is used to decide if there are less than 5 direct interactions, i.e, there are few samples to form correctly the personal opinion on the behaviour of the node.

In Fig. 4.1 we only plot the two strategies for the case of the proactive approach as the choice of the strategy does not modify the amount of information transmitted for the reactive one. The fact that decisions are based only on the opinion value reduces the number of queries to the designated agents, and thus, the overall signalling in the the system.

4.5.2 Reactive and proactive approaches

In Fig. 4.2 we plot the fraction of correct decisions as a function of the number of messages transmitted in the system in order to analyze how the reputation

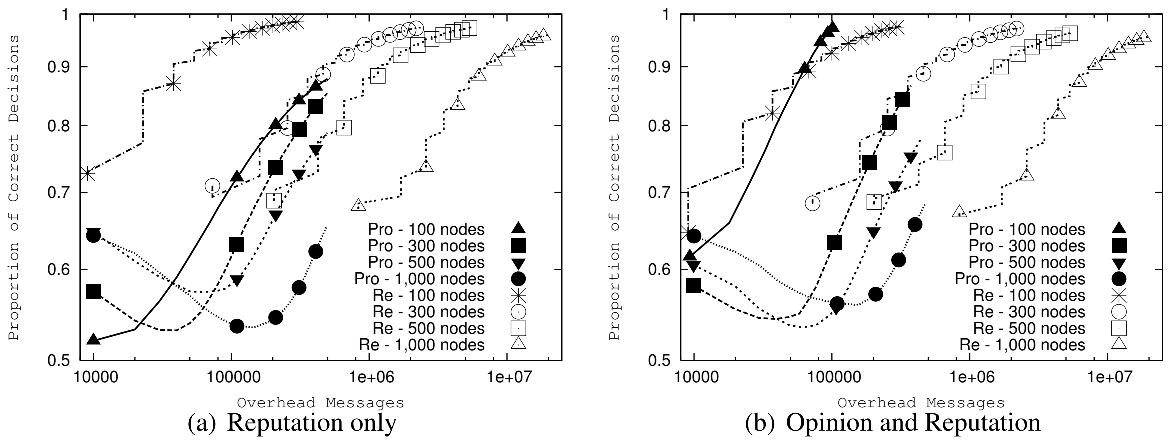
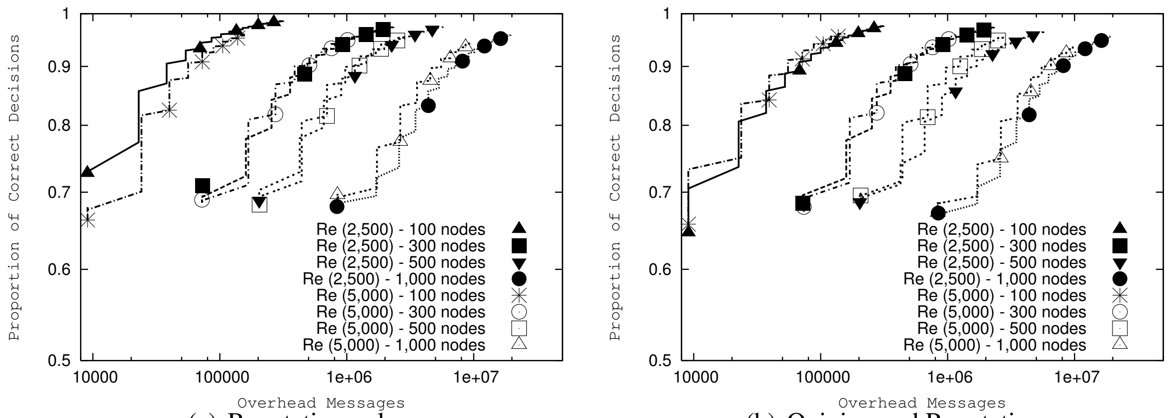


Figure 4.2: Impact of the number of nodes on the proportion of correct decisions and on the number of overhead messages. Collection and dissemination is performed either proactively (*Pro*) or reactively (*Re*) every $2,500$.

management system performs when data are available. The curves show that the number of nodes has a great impact on the performance. When the system is composed of few nodes, a smaller number of messages is sent and the success rate slightly increases. In particular, the reactive approach shows that the amount of overhead is proportional to the number of nodes. On the contrary, the proactive approach stresses that an accurate estimation of the behaviour of the nodes is a function of the number of the interactions. When there are few nodes in the system, nodes interact more frequently, thus, there are more samples to estimate the reputation value of the node.

Fig. 4.2 confirms our hypothesis, as the curves for the proactive approach show that the success rate decreases initially and then increases, as more information is available for the reputation management system. This is more evident when the number of nodes increases.

In Fig. 4.2, we plot two possible choices for the node: use only reputation (a) or opinion first and then reputation (b) if few samples are available for opinion, i.e., less than 5 direct interactions. As evident from the curves in Fig. 4.1, the proactive approach performs better for a small number of nodes in the system in terms of proportion of correct decisions with respect to the number of overhead messages, when opinion is used first and then reputation, plot (b) in Fig. 4.2. This is more evident when there are few nodes in the system as the frequency of the interactions between the same pair of nodes is higher.



(a) Reputation only (b) Opinion and Reputation

Figure 4.3: Impact of the system size, number of nodes, and polling frequency ($2,500$ or $5,000$) for collection and dissemination on the success rate and on the number of overhead messages when the reactive (*Re*) approach is used.

Fig. 4.3 shows the performance of the reactive approach with different values of the frequency for feedbacks collection and dissemination of reputation values. We consider the case of $2,500$ and $5,000$ transactions between two subsequent polls. With a frequency of $5,000$ transactions, the amount of new information, in terms of feedbacks, is higher, thus, the reputation system is able to have a more accurate estimation of the behaviour of the nodes. The system performs better with the same number of messages if we use either reputation or opinion and then reputation to decide the behaviour for a transaction, as shown in Fig. 4.3, (a) and (b) respectively.

However, with few nodes in the system (100 nodes in Fig. 4.3), a more frequent collection of feedbacks has the advantage of updating the reputation value faster and the evaluation of the reputation value is more accurate. As a guideline for the designer of a reputation management system, the time for the poll must be decided based on the number of nodes and on the number of transactions between two subsequent polls.

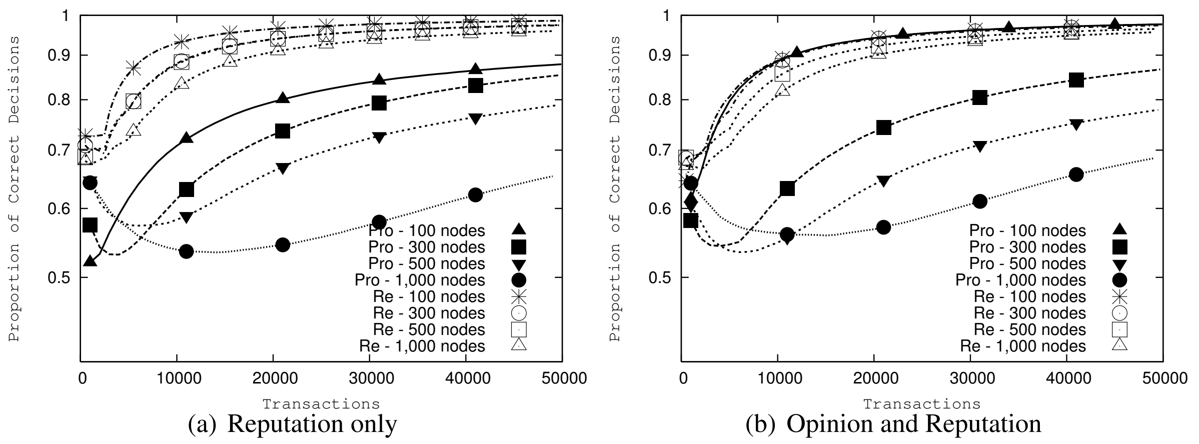


Figure 4.4: Performance of reactive (*Re* - polling frequency is every $2,500$) and proactive (*Pro*) approaches in terms of success rate.

In Fig. 4.4, we plot the success rate as a function of the number of iterations for the proactive and reactive approach; an iteration corresponds to a transaction between to nodes. We can conclude that the reactive approach introduces higher overhead, but it gives better performance as all the nodes have global knowledge in the system. As observed earlier, when there are few nodes in the system, the fact of having frequent interactions between the same nodes means that more samples are available to estimate locally the behaviour of a node. In fact Fig. 4.4 shows that a proactive scheme has a comparable success rate to a reactive approach when there are few nodes in the system and opinion is used first to determine the trustworthiness of the correspondent node.

4.5.3 Number of designated agents

To complete the evaluation of the impact of reputation management systems to the performance of a general application, we analyze how the number of designated agents is important for the overall computation of the communication overhead. Fig. 4.5 plots the success rate as function of the number of messages transmitted in the case of proactive and reactive (every $2,500$ transactions) feedbacks collection and reputation dissemination. It is worth noticing that when only reputation is used as a metric for the trustworthiness of the nodes, a higher number of designated agents is important to predict better the behaviour of the nodes, plot (a) in Fig. 4.5.

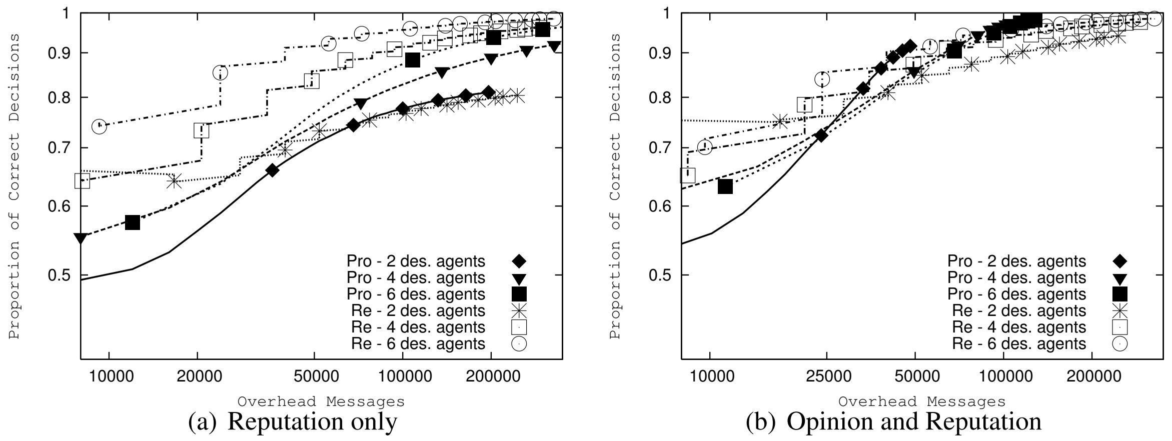


Figure 4.5: Impact of the number of designated agents (2, 4 or 6 des. agents) on the success rate and overhead messages for a system with 100 nodes when the reactive (*Re* - polling frequency is every $2,500$) or the proactive (*Pro*) approach is used.

In our simulation, we consider the case of designated agents that can also be malicious. Thus, multiple designated agents for the same node means more estimations of the node's reputation. This guarantees that the malicious behaviour of a designated agent in reporting reputation values does not influence the evaluation of the trustworthiness of the node. However, the cost of multiple designated agents is paid in terms of number of messages the reputation system uses to collect and disseminate the data.

Similar conclusions are derived from the case of opinion and reputation, plot (b) in Fig. 4.5. In this case, the fact that nodes use opinions to take decisions on a transaction reduces the need for reputation values in the proactive approach and the impact of malicious designated agents also for the reactive approach.

Thus, we can conclude that it is true that, also in this case, the success rate increases when more designated agent are used, but the gain is lower compared to the use of only reputation, plot (a) in Fig. 4.5.

4.5.4 Estimation of reputation values

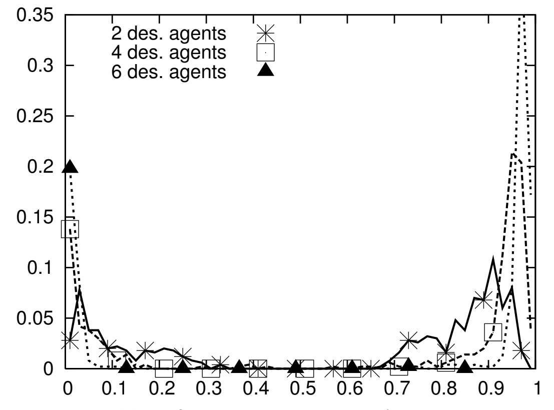
An important metric to evaluate the performances of the proactive and reactive approaches is to analyze the probability distribution of the reputation values in the system.

As discussed in previous sections, varying the number of score managers and the number of nodes in the system has a great impact on the success rate of the reputation management scheme. The accuracy of the reputation values' estimation also depends on these parameters.

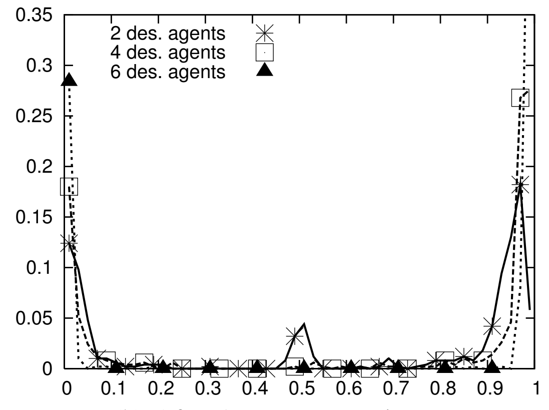
In order to be effective, a reputation management system should score the nodes so that malicious nodes are rated with low reputation and high reputation is associated to cooperative nodes. In Fig. 4.6 – 4.11 we plot the distribution of the reputation values for 2, 4 and 6 designated agents when 30% of the nodes are malicious. To be effective, the distribution of the reputation values should partition the system in two groups: 30% of the nodes with low reputation, i.e., close to 0, and 70% of the nodes with high reputation, i.e., close to 1.

In Fig. 4.6 – 4.8 we consider a system with 100 nodes and in Fig. 4.9 – 4.11 to analyze the performance of the proactive and reactive approaches.

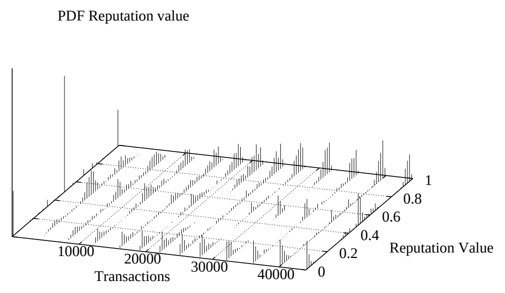
As discussed in Section 4.5.3, a high number of score managers is required to mitigate the effect of malicious agents. Fig. 4.6 shows that when 2 designated agents are assigned to each node (c), trustworthy nodes cannot be rewarded for their cooperative behaviour when a proactive approach is used. However, with 6 designated agents, malicious nodes are identified already after $20,000$ transactions, see plot (a) in Fig. 4.6. We can notice that the population is divided in two groups, malicious nodes (on the right of the plot) and trustworthy nodes (on the left of the plot).



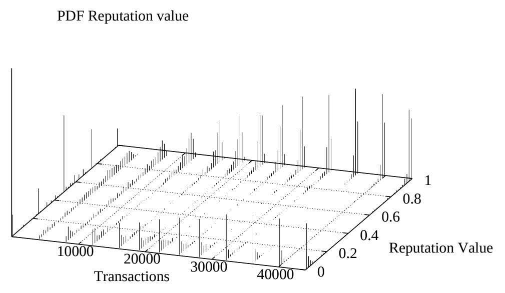
(a) After $22,000$ transactions



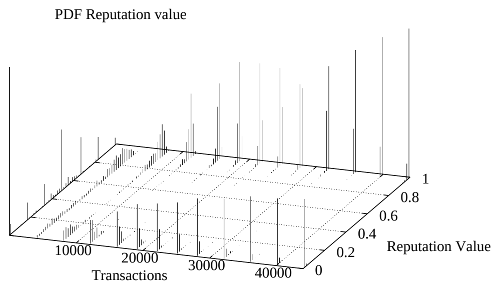
(b) After $45,000$ transactions



(c) 2 Designated Agents



(d) 4 Designated Agents



(e) 6 Designated Agents

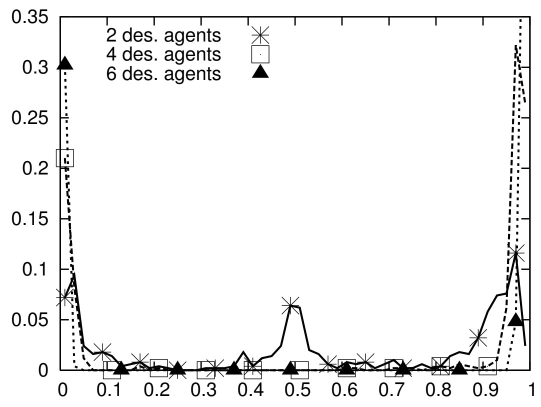
Figure 4.6: Probability Distribution Function (PDF) of the reputation values for a system composed of 100 nodes with a proactive approach for feedback collection and dissemination of reputation values.

In Fig. 4.7, we plot the distribution of the reputation value of the nodes when information is collected and disseminated reactively. In this case, designated agents poll nodes every $2,500$ iterations.

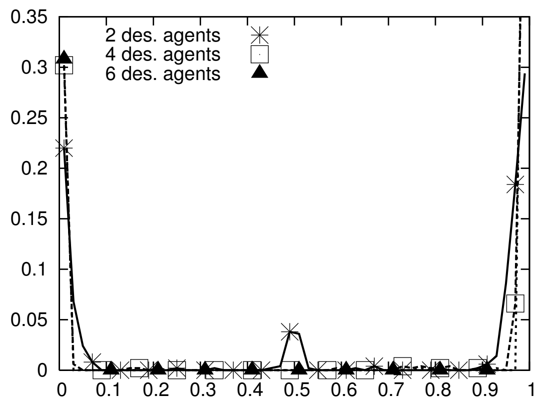
As anticipated in Section 4.5.2, the reactive approach converges faster for the prediction of the nodes' trustworthiness than a proactive one. In particular, 4 designated agents per node are sufficient to give a good estimation of the reputation values. This is due to the fact that when nodes send their feedbacks to designated agents, the feedback is more accurate as it is built on a higher number of direct transactions. The accuracy in the feedbacks enables the score managers to predict better reputation values. Thus, a false report from a malicious designated agent is detected timely and its credibility is lowered.

In Fig. 4.8, the frequency of polling for the reactive approach is every $5,000$ transactions. For this setting, we might expect better accuracy compared to the previous case as nodes form the feedback on a grater number of transactions.

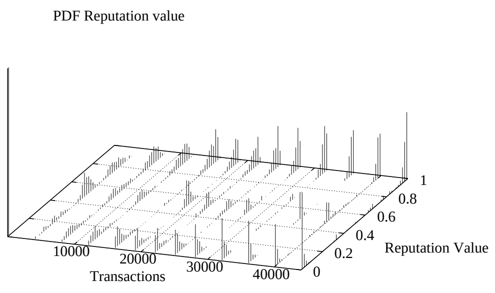
But, the explanation is not completely valid for $5,000$ as the designated agents query nodes less often. As a result, for their decisions nodes use reputation values that does not map the current state of the system. Thus, they might decide to interact with a node based on information that is old. This results in a slower convergence of the reputation management scheme. Moreover, this uncertainty creates room for malicious designated agents to bias the decisions of the nodes.



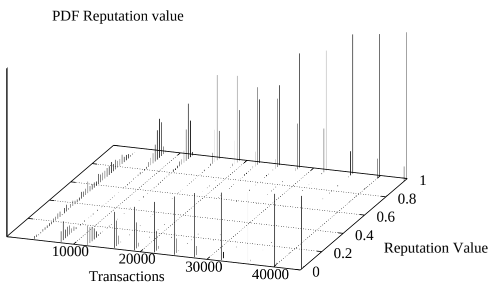
(a) After $22,000$ transactions



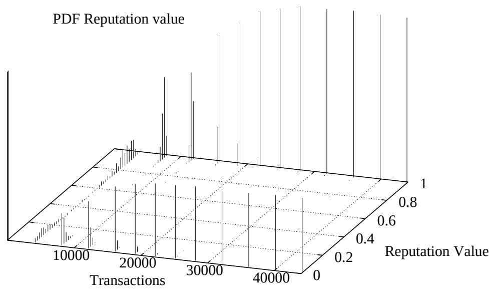
(b) After $45,000$ transactions



(c) 2 Designated Agents

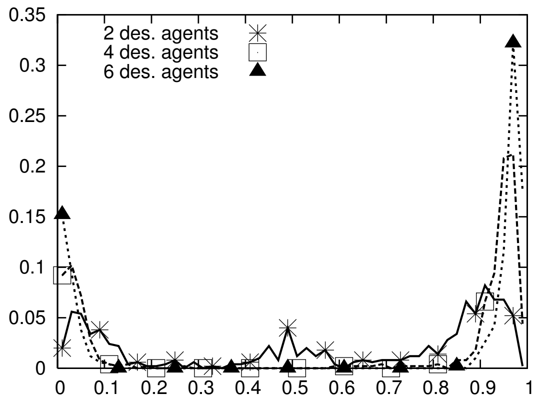


(d) 4 Designated Agents

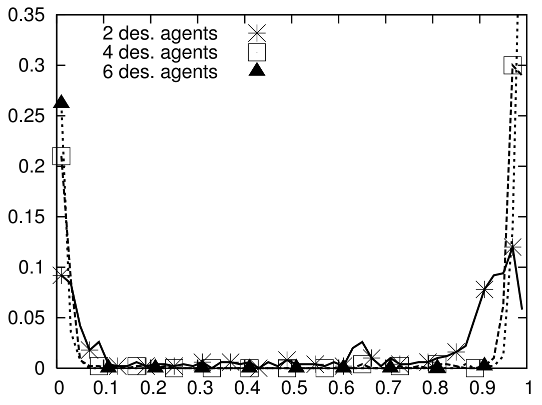


(e) 6 Designated Agents

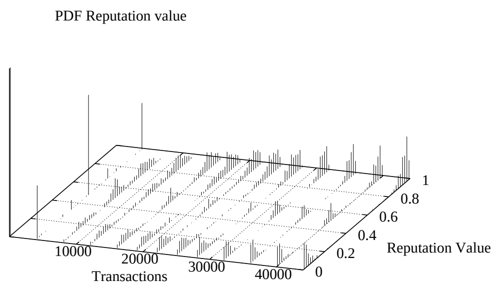
Figure 4.7: Probability Distribution Function (PDF) of the reputation values for a system composed of 100 nodes with a reactive approach for feedback collection and dissemination of reputation values every $2,500$ transactions.



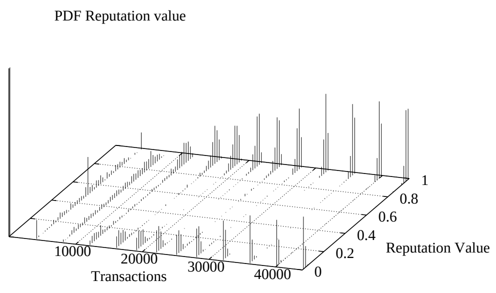
(a) After $22,000$ transactions



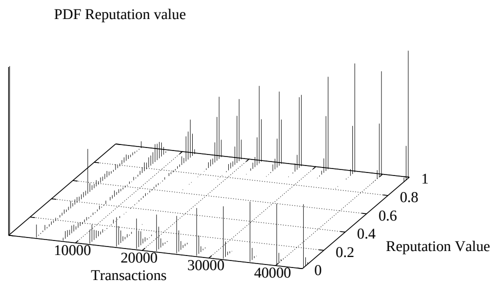
(b) After $45,000$ transactions



(c) 2 Designated Agents



(d) 4 Designated Agents



(e) 6 Designated Agents

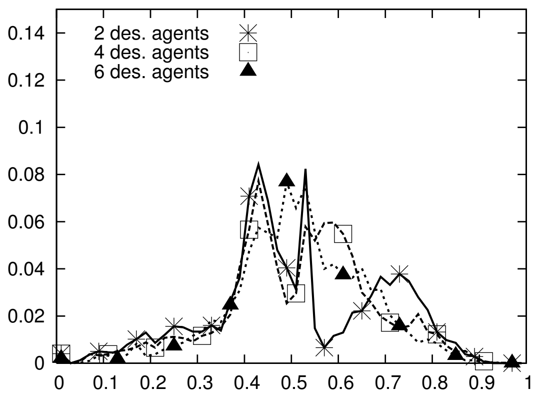
Figure 4.8: Probability Distribution Function (PDF) of the reputation values for a system composed of 100 nodes with a reactive approach for feedback collection and dissemination of reputation values every $5,000$ transactions.

In Fig. 4.9 – 4.11 we show the distribution of reputation values for $1,000$ nodes in the system. The number of nodes influences the accuracy of the reputation values' estimation and the time required to convergence of the reputation management scheme.

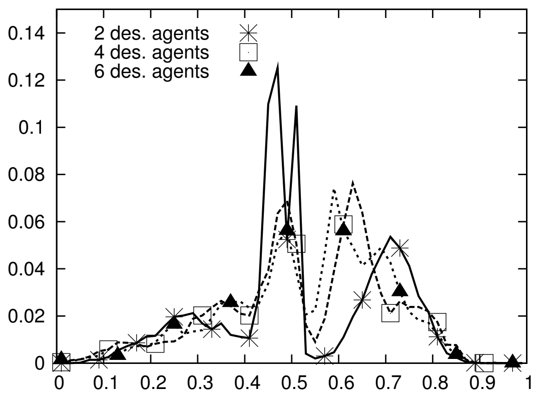
We can notice that when a proactive approach is used for collection and dissemination, the reputation scheme is not able to differentiate between trustworthy and malicious nodes, Fig. 4.9. This is the result of sporadic transactions between the user and the availability of few samples per node to predict the reputation value. This uncertainty is clearly shown in plots (a) and (b) of Fig. 4.9 as the reputation values of the nodes are all almost close to 0.5. In this case, multiple score managers per node do not improve the performance as the impact of the maliciousness of designated agents is covered by the few direct transactions.

From Fig. 4.10 and 4.11, we can conclude that the increase in the number of nodes causes a slower convergence for the estimation of the reputation values, as shown in plots (d) and (e). We recall that when a reactive approach is implemented the feedbacks are collected at regular intervals as clearly shown in plots (d) and (e) of both Fig. 4.10 and 4.11.

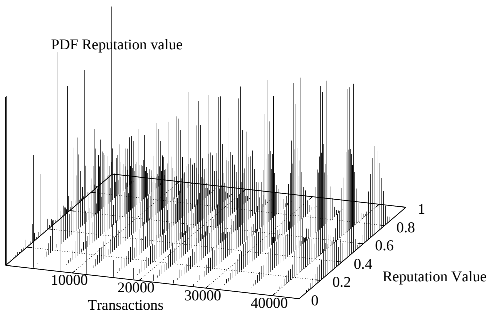
For the reactive approach, it is possible to notice that the number of designated agents is important to evaluate reputation values correctly. Plots (a), (b) and (c) of Fig. 4.10 and 4.11 show that 2 designated agents are not sufficient to make the reputation scheme to function properly. The malicious behaviour of the designated agents increases the uncertainty, given by few samples, for the estimation of reputation.



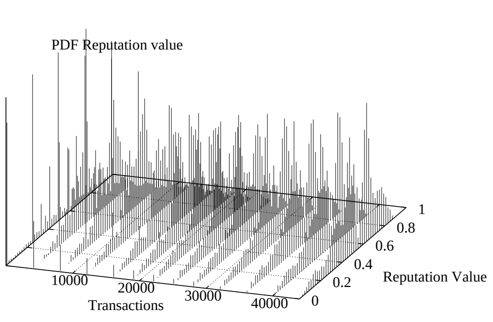
(a) After $22,000$ transactions



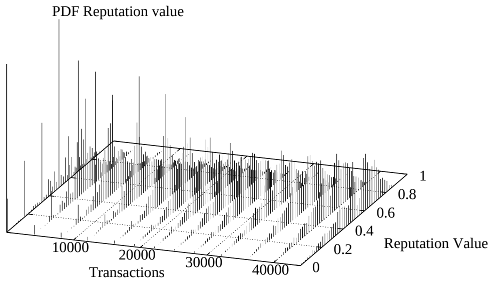
(b) After $45,000$ transactions



(c) 2 Designated Agents

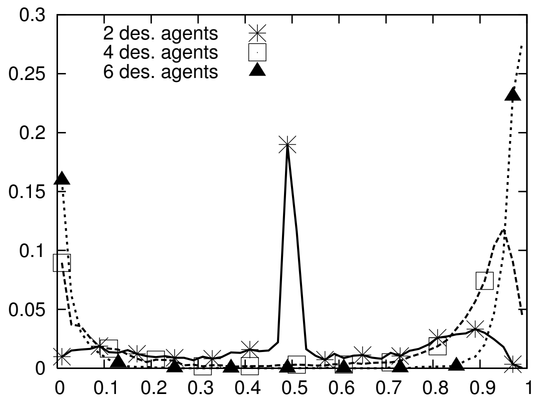


(d) 4 Designated Agents

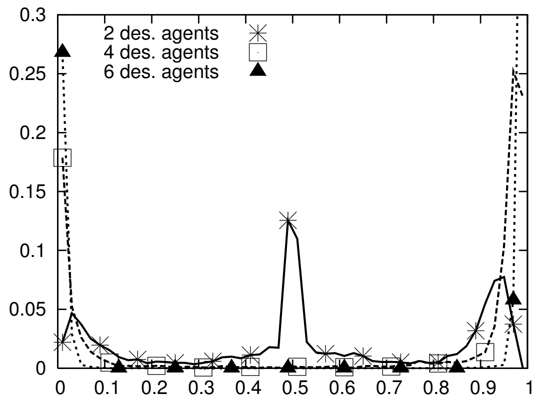


(e) 6 Designated Agents

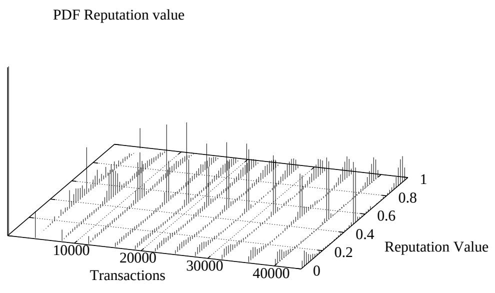
Figure 4.9: Probability Distribution Function (PDF) of the reputation values for a system composed of $1,000$ nodes with a proactive approach for feedback collection and dissemination of reputation values.



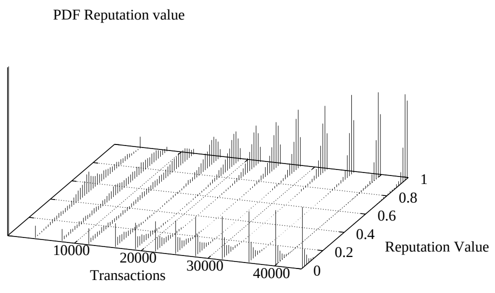
(a) After $22,000$ transactions



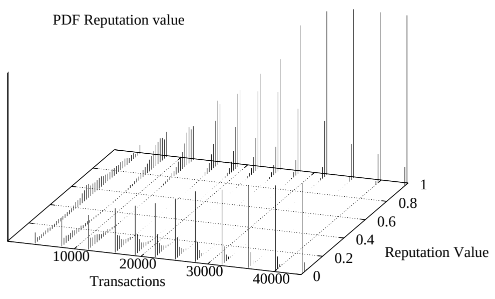
(b) After $45,000$ transactions



(c) 2 Designated Agents

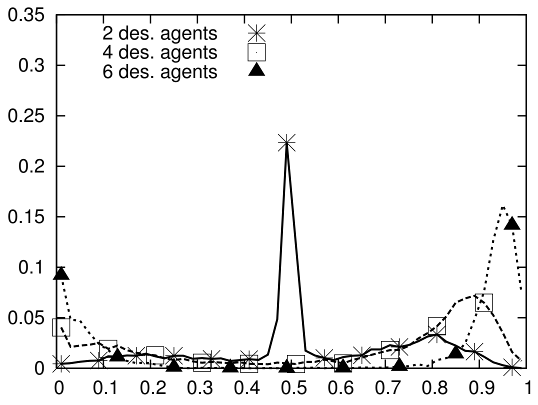


(d) 4 Designated Agents

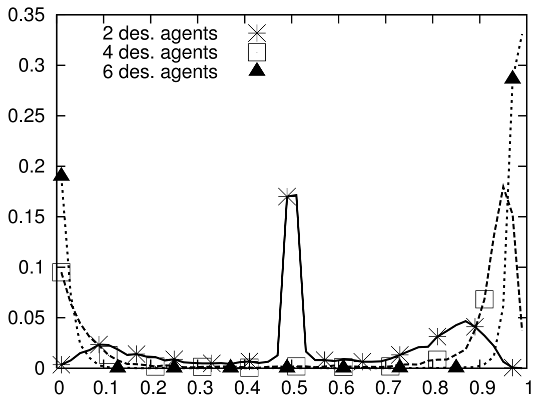


(e) 6 Designated Agents

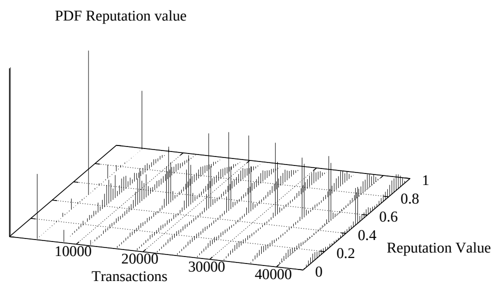
Figure 4.10: Probability Distribution Function (PDF) of the reputation values for a system composed of $1,000$ nodes with a reactive approach for feedback collection and dissemination of reputation values every $2,500$ transactions.)



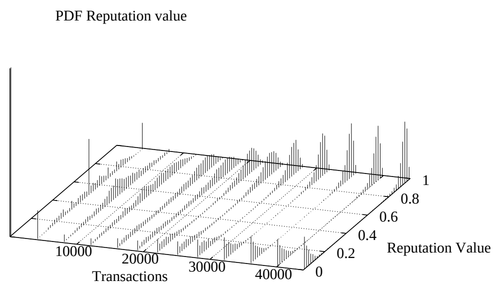
(a) After $22,000$ transactions



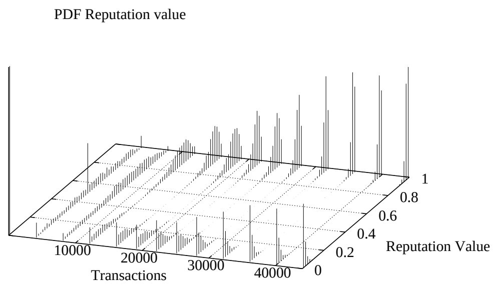
(b) After $45,000$ transactions



(c) 2 Designated Agents



(d) 4 Designated Agents



(e) 6 Designated Agents

Figure 4.11: Probability Distribution Function (PDF) of the reputation values for a system composed of $1,000$ nodes with a reactive approach for feedback collection and dissemination of reputation values every $5,000$ transactions.

Chapter 5

Game theoretical modeling and analysis

5.1 Introduction

Autonomic communication systems are characterized by heterogeneous components that interact to achieve their specific goals and to sustain the scalability of the system. In these systems, service provision and maintance of the network rely on nodes' willingness to fulfill their obligations.

In un-managed and fully distributed systems, incentive mechanisms are required to foster cooperation among nodes. Cooperation upon previous successful experience (see Figure 5.1 (a)) is applicable if the same nodes interact frequently during their lifetime. However, such approach proves to be ineffective as nodes sporadically meet. Another approach is based on indirect collaboration of nodes as shown in Figure 5.1 (b), which implies that collaboration can be established if the transactions are monitored and the result of this observation is shared among the nodes [77].

To foster cooperation, solutions based on service differentiation have been proposed: nodes that contribute more will get better services [58]. This solution leads directly to the adoption of reputation management schemes, defined in Section 3.3. Reputation can be used to give an estimation of the expected cooperation level of the nodes in the view of the community.

As presented in Section 3.6, many reputation schemes have been proposed

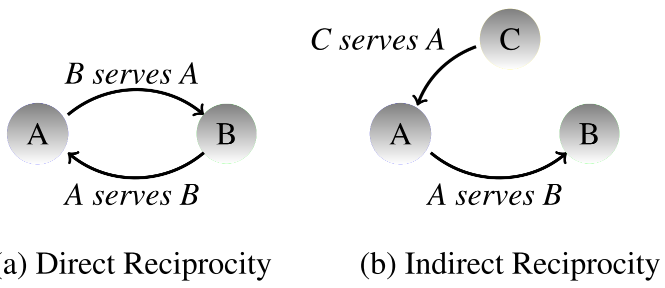


Figure 5.1: Node A serves node B only if it has experienced a positive service in the past, i.e., *direct reciprocity* (a). Node A serves node B to build its reputation so that in the future it can be served by node C, i.e., *indirect reciprociy* (b).

in the literature and have been deployed in different contexts ranging from P2P [49, 63, 8] or content distribution networks [51] to mobile ad-hoc networks [74, 19]. These schemes rely on the dissemination of trust information gathered through transactions between nodes so that past experience can be evaluated and generalized to predict the behaviour of the node. This information is then shared among nodes who can decide whether interact with trustworthy nodes or ignore malicious nodes, as discussed in Chapter 3.

Reputation schemes rely on the definition of a heuristic or aggregation function (see Section 3.4.4) that should capture the nodes behaviour and enable malicious and selfish nodes to be excluded from the system. These schemes are evaluated through simulations experiments, but the lack of a benchmark or set of specific tests to determine their performance in real environment does not help to justify their use in a virtual community.

Moreover, the question how *building* reputation is important for a node's future interactions or how a node *values* its reputation has not yet been clearly addressed. Previous work assumes reputation as a metric to define *cooperation strategy* or to implement a differential service incentive scheme. This is not sufficient to explain why a node should increase its reputation value and to reason on the adoption of reputation management systems from the nodes point of view.

If we consider rational nodes, in the sense that they strategize to increase their expected utility from the system, there is no clear understanding of the role of reputation in selecting a specific action/strategy. In most cases nodes want to cooperate to keep their trust value above a certain threshold that allows them to consume system resources and to provide as few resources as possible in exchange.

In the literature, Marti and Garcia-Molina [72] propose a game theoretic approach to model auctions with multiple buyers and sellers. They show that a weighted calculation of the reputation discourage nodes to defect. Gupta *et al.* [59] define the service game where nodes decide whether to serve other nodes with a probability that depends on the serving node' reputation. In their work they show that in equilibrium the behaviour of different nodes is similar when they provide services.

In this chapter, we analyze reputation management systems from an interdisciplinary perspective taking in consideration social and economic sciences. In our formalization, we discuss the evolution of the system under the enforcement of a reputation management scheme. Herein, we concentrate on incentives and punishment and propose a trust economic model based on the Iterated Prisoner's Dilemma for P2P systems. We finally derive conclusions from the adoption of specific economic theories to model virtual communities and discuss how cooperation can be enforced in autonomous distributed systems formed by selfish agents.

5.2 Network model

We assume an autonomous peer to peer network formed by selfish and rational nodes who want to maximize their own interests from participating in the system activities. Moreover, we assume that 1) the system population is composed of a fix number of nodes (N) with the same capabilities 2) nodes do not participate in a collusion and 3) the identities of the nodes are fixed during the game. Nodes are defined to be rational and strategic: at each interaction they can choose the action, which will influence the outcome of the system. An action can be either cooperation or defection.

We identify two possible network models that are worth investigating for the applicability of reputation management systems in autonomic communication networks: (1) an interaction between nodes is defined as a simultaneously exchange of services, i.e., the transaction is *symmetric*; (2) interactions can be *asymmetric* as nodes have different roles (there is a node that requests a service and a serving node that decides to satisfy the request).

We analyze the second model: the serving node has to decide on the service provision and the receiving node should decide whether to reward the action of the provider. This results in a *non-cooperative* game, as nodes want to maximize their utility. The game is played in multiple stages and nodes follow a strategy (set of actions). We further assume that the end of the game is not known to the nodes and, thus, it is supposed to run indefinitely and that nodes will be present in the system for the whole duration of the game.

We assume a system where at each stage there is a resource request and requests are satisfied at the same rate during the evolution of the game. At the end of each period of time, the results of the interaction are made available to the system population and the utility functions of the nodes are updated. The utility of a node depends on the resources or services it can access and the cost to provide or obtain them. Thus, it is a function of the actions that other nodes in the system take.

A reputation value is associated with each node; this value is updated after every transaction to keep track of node's behaviour and it is assumed to be common knowledge.

5.3 Definition of non-cooperative games

In this section we introduce key notions in non-cooperative games [53] and the formalism that will be used for the rest of the chapter. We restrict the analysis to games in strategic form and in particular we introduce the Prisoner's Dilemma game that is used to model the reputation system.

A game consists of a set of players $N = 1, 2, ..., n$ and for each player $i \in N$ let S_i be the set of possible actions. The deterministic choice of an action $s_i \in S_i$ is called a *pure strategy* for the player $i \in N$ while the vector $s = (s_1, \ldots, s_n) \in \times_{i \in N} S_i$ is the pure strategy profile or *outcome* of the game. $S = \times_{i \in N} S_i$ is defined as the space of all pure strategy profiles, i.e., the Cartesian product of the pure strategies of the players.

For any strategy profile $s \in S$ and player $i \in N$, the *payoff* of the player i, denoted as $\pi_i(s)$, is defined as function of the strategy profile s. Let the collection of the payoffs of the players be denoted by the vector function $\pi = (\pi_i(s))_{i \in N}$. A strategic game is indicated by the triplet $G = (N, S, \pi)$.

In game theory we assume that the players are rational in the sense that they maximize their utility, which is defined by the payoff π_i, by selecting the best strategy against the strategies of the other players given by the deleted strategy profile $s_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)$ with $s_{-i} \in S_{-i}$. S_{-i} is the space of all pure strategy profiles for all players $j \in N$ with $j \neq i$.

Definition A strategy $s_i^* \in S_i$ is a *best response* for player $i \in N$ to the deleted strategy profile $s_{-i} \in S_{-i}$ iff $\pi_i(s_i^*, s_{-i}) \geq \pi_i(s_i, s_{-i}), \forall s_i \in S_i : s_i \neq s_i^*$.

Definition A *pure Nash Equilibrium* is a strategy profile or outcome $s \in S$ iff $\forall i \in N$, s_i is a best response of player i to the deleted strategy profile s_{-i}. That is $\pi_i(s_i^*, s_{-i}) \geq \pi_i(s_i, s_{-i}), \forall i \in N, \forall s_i \in S_i : s_i^* \neq s_i$.

This definition states that, in a *Nash equilibrium* strategy, no player has incentives to deviate **unilaterally** from its best response strategy.

Table 5.1: Generalized Form of the Prisoner's Dilemma game: payoff matrix. Temptation, Reward, Sucker, Punishment

		Receiver	
		Cooperate	Defect
Serving	Cooperate	(R_s, R_r)	(S_s, T_r)
	Defect	(T_s, S_r)	(P_s, P_r)

Thus, an important property of a system is characterized by the outcome that maximizes players' payoff, i.e., in finding the *Nash equilibrium*. However, multiple Nash equilibria might exist in a game and players might have different payoffs in different Nash equilibria of the game.

The rules of the game are based on two basic assumptions: 1) players choose their own strategies optimally based on the beliefs they have on the other players' strategies; 2) the prediction of other players' strategies must be correct.

5.4 Prisoner's dilemma

We model the autonomous system by using definitions and results based on the Prisoner's Dilemma [53, 78]. In this game two nodes decide simultaneously whether to cooperate or defect without knowing a-priori the choice of the other player. If both cooperate, they receive a specific reward (*R*). If both defect, they receive a punishment (*P*). If one defects and the other cooperates, the player who defects will receive a larger reward, temptation (*T*), and the other will receive a larger punishment, the sucker's payoff (*S*).

Table 5.1 shows the strategic form of the game represented in a bi-matrix where the rows and the columns are the available strategies for the two players and each box specifies the payoff to each player, respectively for player (S)erving and (R)eceiving, when the strategy profile corresponding to that cell is played.

The Prisoner's Dilemma is a non-cooperative and simultaneous game where in its generalized form the payoffs are not identical for the players. To create the dilemma, mutual cooperation must provide higher payoff than other strategies and defection should be the dominant strategy. Therefore, the following conditions must hold:

$$T > R > P > S \tag{5.1}$$

$$R_s + R_r > P_s + P_r \tag{5.2}$$

In the single stage game, the best choice for the players is to defect, i.e., $(Defect, Defect)$, as it is always the best response to the opponent strategy [53]. More interesting is the iterated version of the game where nodes play against each other repeatedly and track the history of the game. In this setting, nodes can be punished for their defection in past interactions.

In the iterated version of the game, the payoff of the players are computed by summing the single stage payoff over all the stages played. However, to maintain the dilemma in the iterated version, the following inequality must also be valid:

$$R_s + R_r > S_s + T_r \text{ and } R_s + R_r > T_r + S_r \tag{5.3}$$

Conditions (5.3) state that alternation between defection and cooperation, i.e., nodes take the action $(Defect, Cooperate)$ and $(Cooperate, Defect)$ in subsequent transactions, does not give higher payoff than mutual cooperation, i.e., always $Cooperate, Cooperate$.

However, in the finite iterated version of the game, rational players defect for their last move as it has higher payoff. But if the game is iterated infinitely (it is sufficient to assume that nodes do not know when the game ends) cooperation results in a Nash equilibrium of the game, i.e., where no player has incentives to deviate **unilaterally** from its best response strategy.

5.5 The reputation game

This section describes the reputation game which is based on the Iterated Prisoner's Dilemma. At each stage of the game two nodes, picked at random, are considered for the game: one node is acting as resource provider and the other as resource consumer. Their roles in the system are interchangeable, i.e., a node who is service provider at time t_i might be service receiver at time t_j. In our model, we use the general term resource as it can represent a file in the case of content distribution networks, a request to cache a file in distributed caching systems, a job execution request and so on.

The two available actions are collaborate (provide the service) or defect (ignore the request) for the serving node and collaborate (reward) or defect (do not reward) for the receiving node. We are interested in showing how the decisions affect nodes utility and reputation values in the long run of the game.

5.5.1 Introducing reputation in the game

First we define the role of reputation in the game. A provider node pays a cost for providing the service, but this cost is compensated by the increase in its reputation I and by the reward obtained from the receiving node. A receiver pays the requested service but it increases its reputation value as a result of fulfilling its commitment.

After each transaction, the reputation value is updated, based on the reputation calculated at the previous stage and of the outcome of the single stage game. The updating function is defined as follow:

$$I_{t+1} = \begin{cases} 0, & \text{if } t = 0 \\ I_t * (1 - \alpha) + v * \alpha, & \text{if } t > 0 \end{cases} \tag{5.4}$$

with $0 \leq \alpha \leq 1$ and $v \in \{0, 1\}$, thus, $0 \leq I_t \leq 1$.

This means that if a node cooperates, it will increase its reputation by a factor determined by the constant α, which models the importance of the new interaction for the computation of the reputation, and v, a binary parameter that indicates if cooperation has taken place. α is a system parameter and depends on network conditions. If transactions are infrequent, a low value of α is desirable whereas when transactions are frequent, a high value is desirable. Nodes with high reputation values are considered trustworthy in the system.

5.5.2 The reputation model

The reputation game evolves as follow, where steps $2.a$ and $2.b$ are simultaneous:

1) The requesting node identified as N_r has an associated reputation value I_r. It sends a request for a specific service to other nodes in the system by offering a reward B. The selection of the service provider is done by selecting the server with the highest reputation I_s.

2.a) The serving node N_s has two possible actions: 1) cooperate and send the service or 2) defect and ignore the request.

2.b) The receiving node N_r has also two possible actions: 1) cooperate and send the promised reward or 2) defect and fail to meet the commitment.

3) The single stage game ends and the nodes update their utility functions and reputation values.

The dilemma for N_s consists of deciding to:

a) afford the cost of serving the file and behave correctly, thereby obtaining the reward R_s if N_r cooperates or the punishment S_s if N_r defects or;

b) ignore the request obtaining the reward T_s without having sent the file if N_r fulfills its commitment or the punishment P_s if N_r defects as well (see Table 5.1).

N_r has to face a similar dilemma but with different payoffs.

The payoffs for a generic serving node N_s and requesting node N_r at a specific iteration of the game t are given by eq. 5.5 and eq. 5.6.

$$\pi_s^{t+1} = \begin{cases} -C + B - C_p(I_r) + f(I_s^t), & R_s \\ -C - C_p(I_r) + f(I_s^t), & S_s \\ B + f(I_s^t), & T_s \\ f(I_s^t), & P_s \end{cases} \tag{5.5}$$

$$\pi_r^{t+1} = \begin{cases} -B + S + g(I_r^t), & R_r \\ -B + g(I_r^t), & S_r \\ S + g(I_r^t), & T_r \\ g(I_r^t), & P_r \end{cases} \tag{5.6}$$

where C is the cost for providing the service, B is the reward for serving nodes and S is the value of a service for the requesting node. $C_p(I_r) = B/(1 + e^{5I_r})$ is the punishment factor and $f(I_s^t) = B * [I_s * (1 - \alpha) + \alpha * v]$ and $g(I_r^t) = S * [I_r * (1 - \alpha) + \alpha * v]$ are the benefit for the nodes in terms of *future payments* based on their level of cooperation, respectively for the serving and receiver node (refer to (5.4) for the update function of the reputation). The punishment factor $C_p(I_r)$ is inserted to reduce the payoff of serving nodes when providing services to untrustworthy nodes. To provide incentives for cooperation to the nodes, we must have that $(B - C) > 0$, $(S - B) > 0$.

The game is a Prisoner's Dilemma if the conditions introduced in Sec. 5.4 hold. From condition (5.1) we can derive for N_s that $f(I_s)_{\text{def}} > f(I_s)_{\text{coop}} - C_p(I_r) - C$ and $B > C + C_p(I_r)$. This means that the benefit in serving must compensate the direct cost and the cost derived from the punishment of serving less trustworthy nodes.

Hence, the lower the reputation I_r, the greater is the punishment that N_s will receive, thus, it is more tempted to defect. For the requesting node N_r, it is sufficient to have that $g(I_r)_{\text{def}}+B > g(I_r)_{\text{coop}}$ as we have assumed $(S-B) > 0$.

The dilemma is kept in the iterated version of the game (5.3) if $S > C + C_p(I_r)$ as we assume the utility, derived from future payments, be greater in case of cooperation.

5.5.3 Nash equilibrium for the reputation game

In this section we demonstrate the effectiveness of the reputation game and we show how Nash equilibrium is obtained by analyzing the impact of actions at each stage on the reputation value. We use a modified version of the ROCQ reputation management system, defined in Section 3.7, on top of a peer to peer network to simulate nodes interactions. The parameters' settings and decision metrics that we use to evaluate the results, obtained from the simulations, are summarized in Table 5.2 and they are defined in accordance with the conditions of the Iterated Prisoner's Dilemma, introduced in Sec. 5.4.

Table 5.2: Parameters' setting

Number of Nodes	1,000	Network Topology	random
Initial transactions	1,000	Number of iterations	9,000
Experiments run	5	Service value [S]	25
Benefit [B]	15	Cost [C]	5

In the repeated game strategy, reputation is sufficient to sustain cooperation in the system if the players are patient. Let's consider the repeated game with a trigger strategy where the serving node N_s and the receiving node N_r cooperate for t transactions and at transaction $t+1$ the provider node N_s *unilaterally* de-

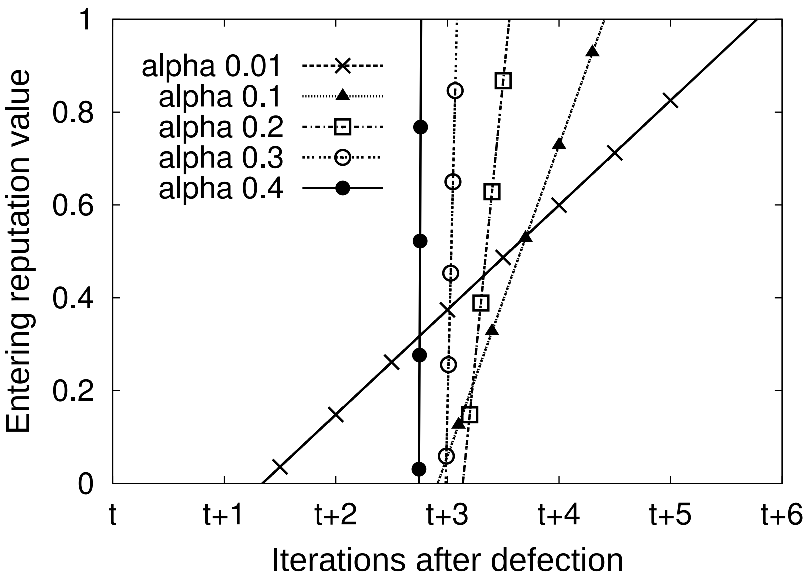


Figure 5.2: Nash Equilibrium for the reputation game: the defecting player has greater payoff from always cooperation ($\pi = \sum R_i$) rather then cooperation defection ($\pi = \sum^t R_i + T_{t+1} + \sum_{t+2} P_i$) (this is represented by the points below the curves when $t = 10$ is the last cooperative interaction.) The Nash Equilibrium depends from the reputation value at the beginning of the game.

fects triggering an-open loop for N_r to defect in subsequent interactions. After transaction $t+1$, the *best response* for N_s is to defect.

The curves in Fig. 5.2 show when the payoff resulting from this action is equal to the payoff earned from cooperating all the time. This is summarized in eq. (5.7) that indicates after how many x interactions, with x defined after transaction t, *always cooperate* is the best strategy with respect to the entering reputation value $I_s(t = 0)$.

$$\begin{aligned} I_s(t=0) \leq & \frac{10x + 15(2-\alpha)[(1-\alpha)^t - 1] - 15(1-\alpha)^x}{25(2-\alpha)(1-\alpha)^t} + \\ & + \frac{15/(1 + e^{5*\{(1-\alpha)^t[I_r(t=0)-1]+1\}})}{25(2-\alpha)(1-\alpha)^t} \end{aligned} \tag{5.7}$$

The plot in Fig. 5.2 shows that the defecting player has no incentive for being uncooperative at transaction $t+1$ as the *Temptation* reward obtained for

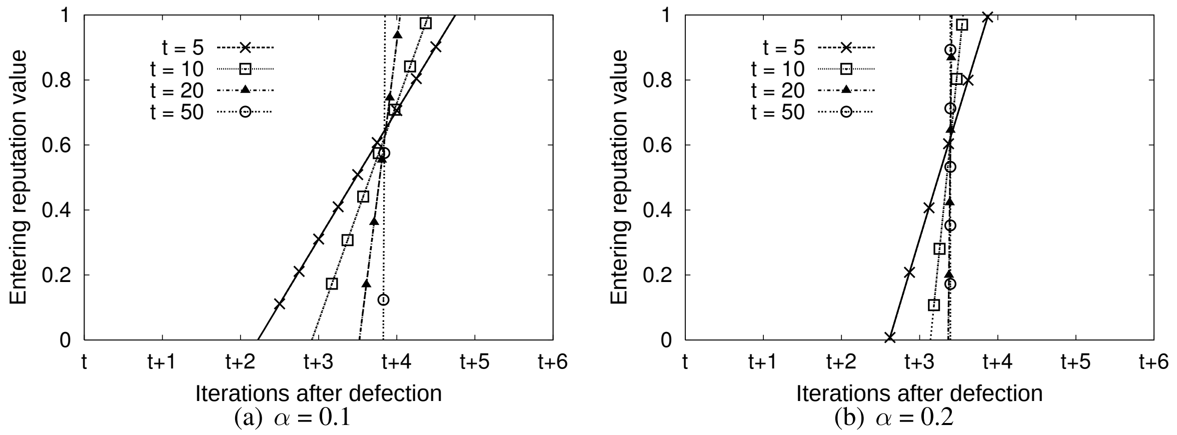


Figure 5.3: Definition of the regions that define when it is more convenient for nodes to always cooperate rather than cooperation defection for different values of t, i.e., the first defecting transaction, with $\alpha = 0.1$ (a) and $\alpha = 0.2$ (b).

this interaction is not sufficient to compensate the loss due to future *Punishment* rewards.

For different choices of the parameter α (used to calculate the reputation value, refer to eq. (5.4) for the updating function) and for the initial reputation value $I_s(t = 0)$, that the defecting node N_s had at the beginning of the game, it is more advantageous to cooperate when $I_s(t = 0)$ is below the curves (defined by eq. (5.7)) as it is the case after 5 interactions (transaction $(t + 6)$), for all considered values of α.

Fig. 5.3 shows that the incentive for being cooperative increases as the number of transactions before defection increases. In particular, when α is small, a player with a high reputation value has a slight benefit when defecting immediately after entering the game, plot (a). For higher values of α, plot (b), the reputation value of the player drops to a low value that characterizes the nodes as untrustworthy in 3 transactions after defection.

Thus, we have established that if nodes are *sufficiently* patient, i.e., they do not maximize immediately their payoff by defecting but they value more future payments, cooperation is a Nash equilibrium for the *reputation* repeated game.Thus, they need to cooperate for few transactions at the beginning of the game to build their reputation inside the community, to benefit from their participation. The same result can be derived for the receiver as defecting node.

Table 5.3: Strategies available to players

Average	A node decides for cooperation if the opponent reputation value is greater than the average reputation.
Adaptive	A node considers its reputation value, the correspondent reputation value and the average reputation in the system to decide for cooperation or defection.
Relative	A node cooperates if its reputation value is below the correspondent reputation value.
Discriminant	A node decides for cooperation if the correspondent reputation value is above a fix threshold (if not specified the threshold is set to 0.5).
Random	A node decides randomly if cooperate (this is the only strategy that is not based on reputation).

5.6 Experimental results

In our experiments we use the parameters listed in Table 5.2 and we run $1,000$ initial transactions to bootstrap the reputation management system. In each stage, two nodes are selected randomly, which can act either as service provider or as service receiver. We follow the strategies defined in Table 5.3 for the game and the nodes follow the same strategy in each experiment.

Fig. 5.4 shows the impact of the different strategies in terms of fraction of cooperative interactions. A cooperative transaction is defined as an interaction when both players are cooperative. In this case, the *discriminant* strategy gives better results compared to the others, but this strategy heavily depends on the threshold for the reputation value chosen to differentiate between cooperation and defection.

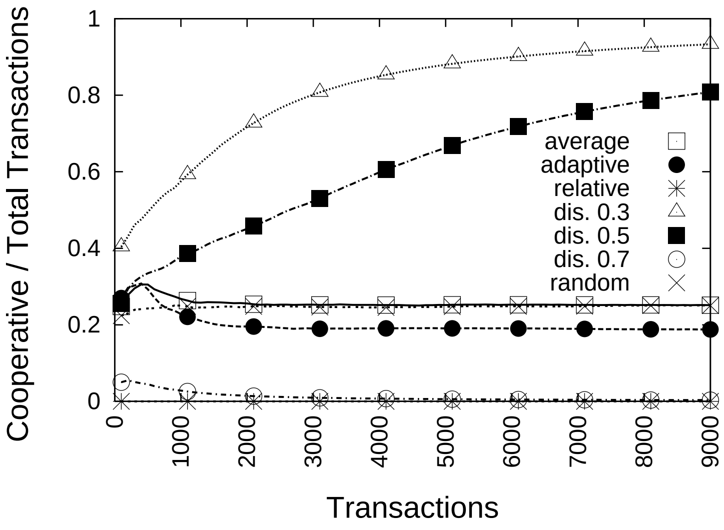


Figure 5.4: Number of cooperative transactions (both provider and receiver cooperate) in the system for the available strategies.

Fig. 5.5 also shows that the discriminant strategy performs better as it increases the average reputation value when it is above the fixed threshold.

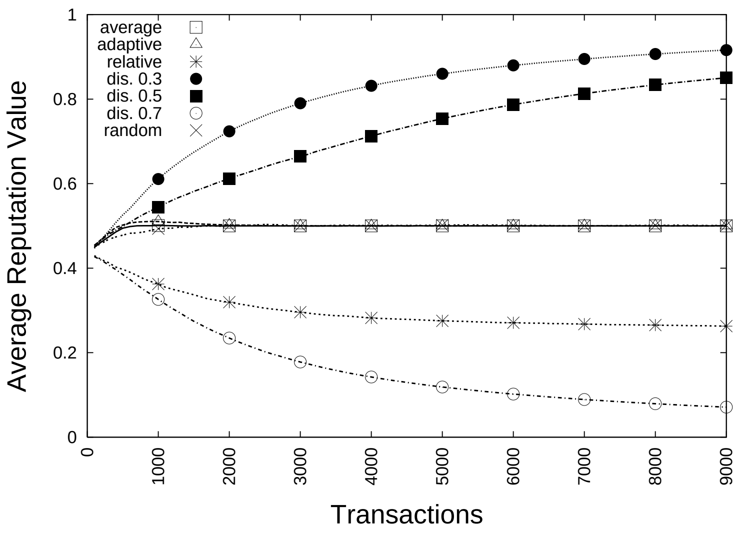


Figure 5.5: Average reputation value in the system for the available strategies.

An interesting property is associated with the average and adaptive strategies as they maintain a constant average trust level (see Fig. 5.5) and constant fraction of cooperative transactions in the system (see Fig. 5.4). The explanation of this behaviour is associated with the definition of the strategies as they choose

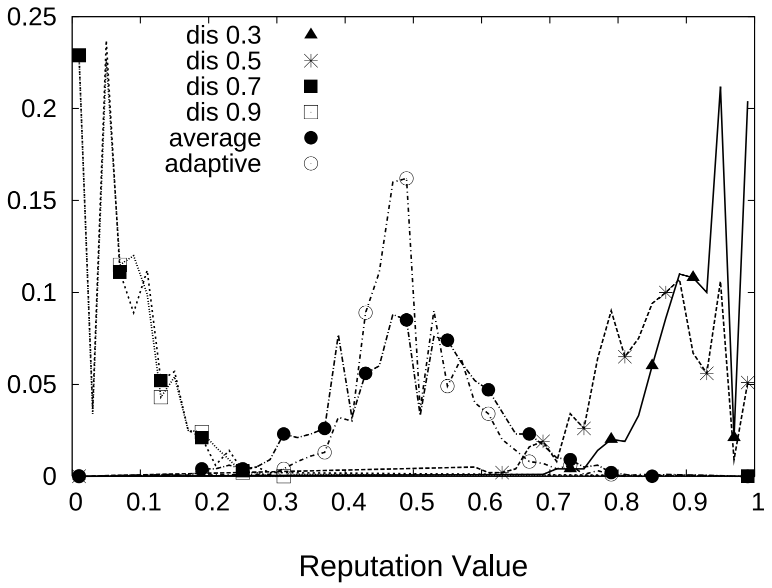


Figure 5.6: Probability Distribution Function (PDF) of reputation values in the system with respect to different strategies.

cooperation or defection to maintain stable the reputation value.

Fig. 5.6 shows the distribution of reputation values in the system. The plots show that the adaptive and average strategies tend to aggregate the reputation values of the nodes close to 0.5, as anticipated above. For the case of the average strategy, the threshold used to decide for cooperation or defection is the average trust value in the system. Thus, nodes choose cooperation and defection alternatively to keep their reputation value close to the average. The discriminant strategy with a low threshold 0.3 shows a behaviour that can be assimilated to always cooperate. The effectiveness of all strategies depends on the reputation value of the nodes after bootstrapping the reputation management system.

Fig. 5.7 shows the impact of the initial reputation value $I(t = 0)$ after the bootstrap of the reputation management system. Fig. 5.7 plot the average reputation values in the system for an initial value close to 0.7 and 0.3. As expected, the discriminant strategy depends on the threshold that a node chooses to decide whether to cooperate or defect. It is worth noticing that cooperation is not propagated in the system, if the threshold is above the initial reputation value.

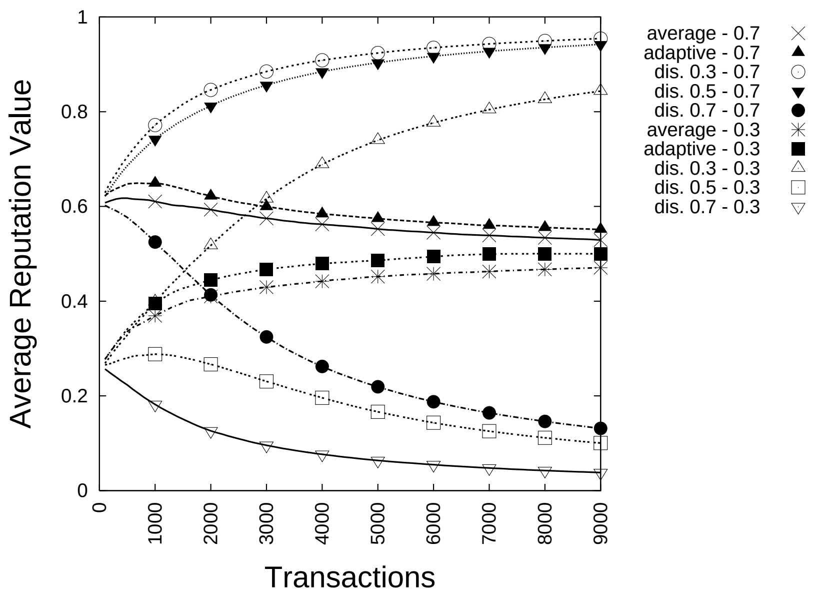
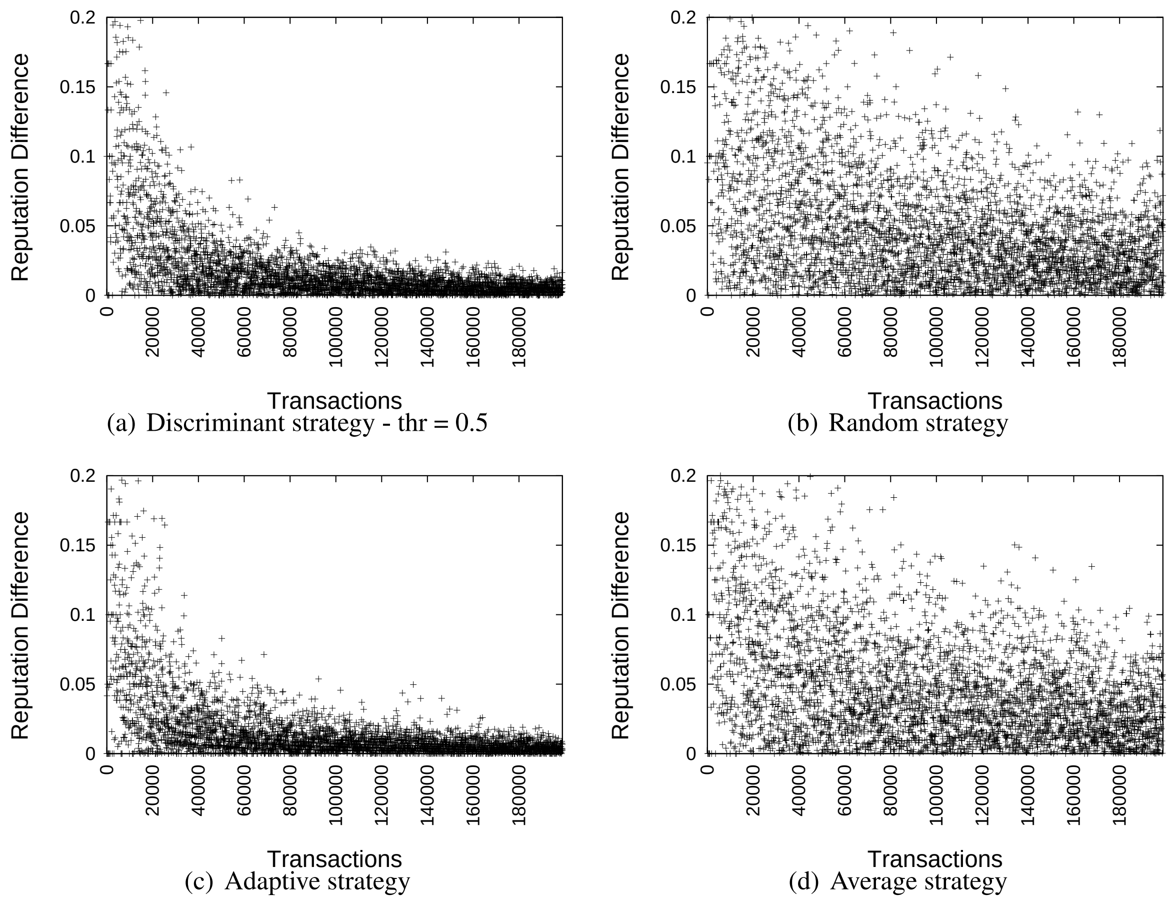


Figure 5.7: Average reputation value considering different conditions of the system at the bootstrap. Initial training transactions are cooperative with a probability of 0.7 or 0.3.

An interesting property is associated with the adaptive and average strategies as they work to bring the system to a stable operating point that has the average reputation value close to 0.5.

Fig. 5.8 plot the difference of the reputation values of the nodes involved in a transaction. We ran the simulation for $200,000$ interactions to study if the reputation values of the nodes converge to the same value. The plots for the discriminant strategy (a) and the adaptive strategy (c) show that nodes tend to interact with nodes that have similar reputation values. As we have assumed that the nodes follow the same strategy in a simulation, we can derive conclusions on the convergence of the system to a stable point for the nodes' reputation.

In Fig. 5.6, it is shown that nodes tend to have a reputation value close to 0.5 for the adaptive strategy. Thus, the node strategies to have a reputation value close to the mean value, for this reason we call this strategy *adaptive*. On the contrary, the discriminant strategy fosters cooperation in the system, as the average reputation value increases in Fig. 5.5, and the the reputation values is close to 1 in Fig. 5.6.



(a) Discriminant strategy - thr = 0.5

(b) Random strategy

(c) Adaptive strategy

(d) Average strategy

Figure 5.8: Difference between the reputation values of the peers involved in a transaction.

Thus, we can conclude from plot (a) in Fig. 5.8 that this strategy tends to aggregate the nodes to similar reputation value and it is a good solution to combat the problem of free-riders in autonomic communication systems if there are no malicious nodes and the initial reputation value is sufficient to foster cooperation in the beginning.

The nodes that select the random and the average strategies do not converge to a stable reputation value, i.e., a consistent behaviour in the system. They tend to alternate cooperation and defection without giving any guarantee to the correspondent party which cannot predict the next action with accuracy.

5.7 Impact of defecting nodes

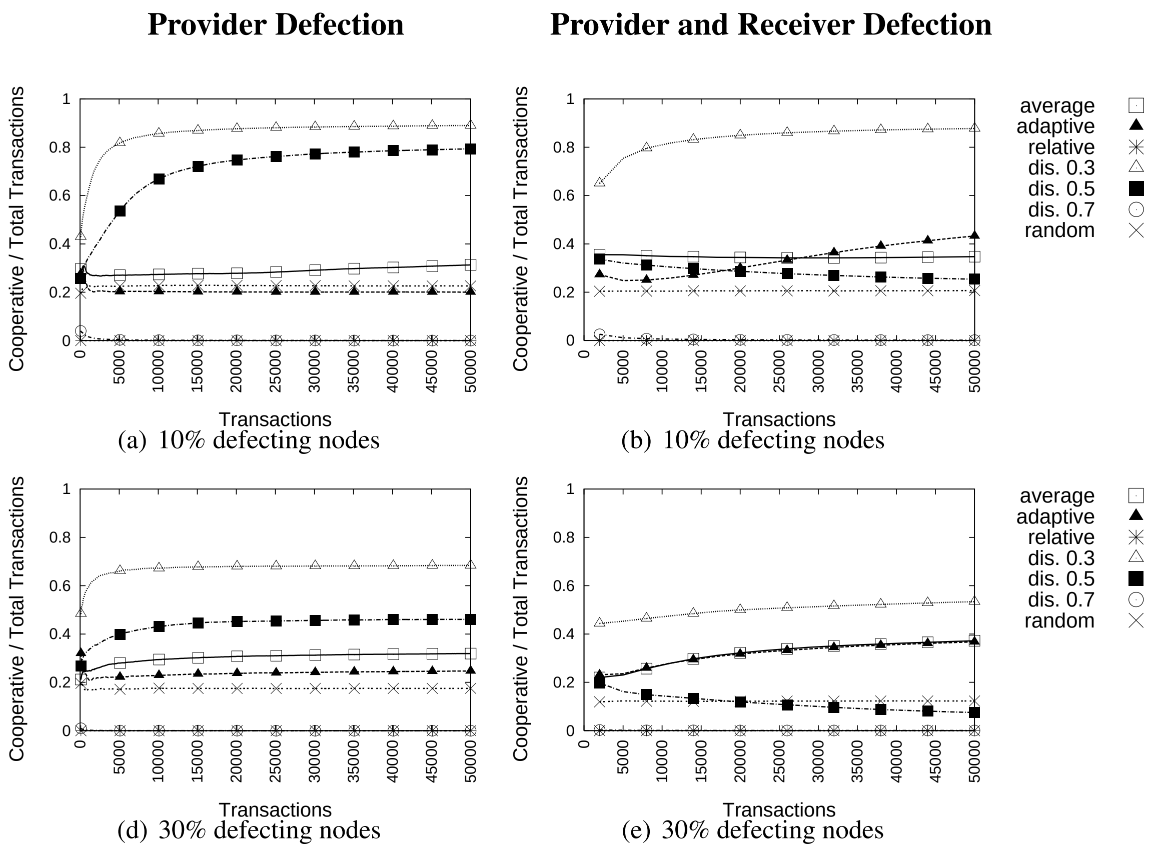
In this section, we analyze the impact of defecting nodes on the strategies discussed in previous sections. We define two classes of defecting nodes to model *free-riders* and completely *uncooperative* nodes which prefer not be involved in any transaction.

Free-riders cooperate when they request services, but they deny access to resources when they receive a request. They are not interested in increasing their reputation value, but only in accessing the wanted resource and fulfilling the obligation with the service provider. Uncooperative nodes deny the access to resources and they do not pay back the service to the provider.

Fig. 5.9 shows the fraction of cooperative transactions, when both players cooperate, with 10% and 30% of free-riders (plots (a) and (d) respectively) and of uncooperative nodes (plots (b) and (e)) in the system. A small percentage of free-riders (a) do not degrade the performance of the reputation system as their partial uncooperative behaviour is compensated by trustworthy nodes. They actively participate in transactions when they are receiver, which also counts for the total number of cooperative transactions. If we increase the percentage of nodes (d) that are only interested in receiving resources, the fraction of cooperative transaction decreases as there are less nodes that are willing to contribute resources.

The impact of uncooperative nodes reduces significantly the number of cooperative transactions as any attempt to interact with these nodes is useless. Only the adaptive and the average strategies are able to operate well in this settings, plot (e) in Fig. 5.9, as the decision for cooperation or defection is based on the average reputation value.

Our hypothesis is confirmed by Fig. 5.10, as the average reputation value tend to be close to 0.5. This value implies high uncertainty for the evaluation of nodes' behaviour and introduces a higher risk in transactions. This risk is given



(a) 10% defecting nodes (b) 10% defecting nodes

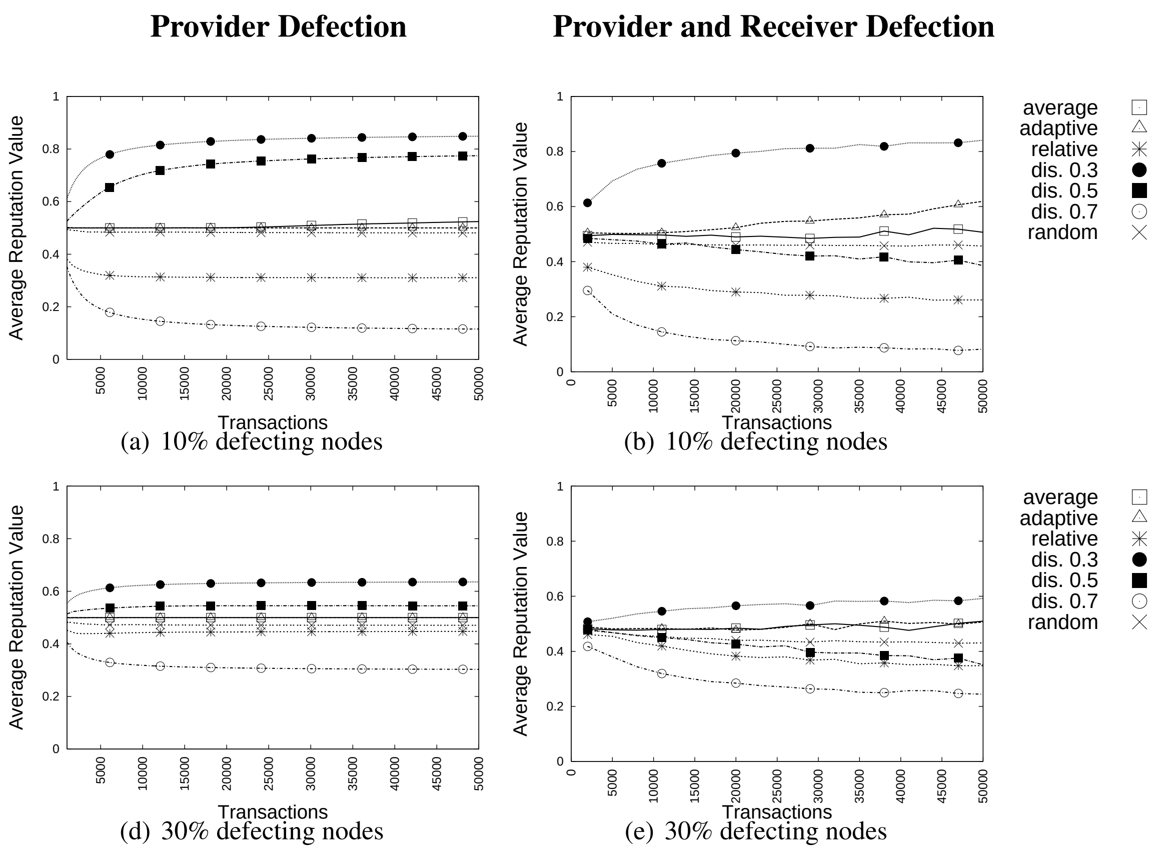
(d) 30% defecting nodes (e) 30% defecting nodes

Figure 5.9: Fraction of cooperative transactions (both provider and receiver cooperate) for the available strategies.

by the fact that nodes are not able to determine if the correspondent party is a free-rider or a uncooperative node.

The lack of accuracy for the prediction of the reputation value is more evident in Fig. 5.11. In this figure, we plot the probability distribution of the reputation values for the nodes in the system. In an ideal situation, two classes of nodes should be present in the system, cooperative nodes, with high reputation values, and uncooperative nodes, low reputation values. This is not our case as reputation values are almost uniformly distributed except for the threshold strategy, plots (b) and (d) in Fig. 5.11. This is the result of the behaviour of defecting nodes that bias the strategies' performance of cooperative nodes.

Plots (a) and (d) of Fig. 5.11 show the presence of a group of nodes that



(a) 10% defecting nodes (b) 10% defecting nodes

(d) 30% defecting nodes (e) 30% defecting nodes

Figure 5.10: Average reputation value in the system for the available strategies.

have similar reputation values for the discriminant strategy. The reason behind this behaviour is due to the fact that free-riders cooperate when they are service receivers, thus, they increase their reputation value. However, their cooperation is only in one direction and this causes the system to lower their reputation since they are not rated as trustworthy nodes. If we increase the fraction of free-riders, the distribution of reputation values show that a high fraction of nodes has a reputation value between 0.4 and 0.6. This is true for all strategies plotted in Fig. 5.11 except for the discriminant strategy with thresholds of 0.3 and 0.7.

This behaviour is consequence of the amount of risk node put in their interactions. For a discriminant strategy with threshold 0.3, nodes are willing to take higher risk in their transaction and this risk is compensated by a greater number of cooperative transaction, see Fig. 5.9, which reflects the high reputation

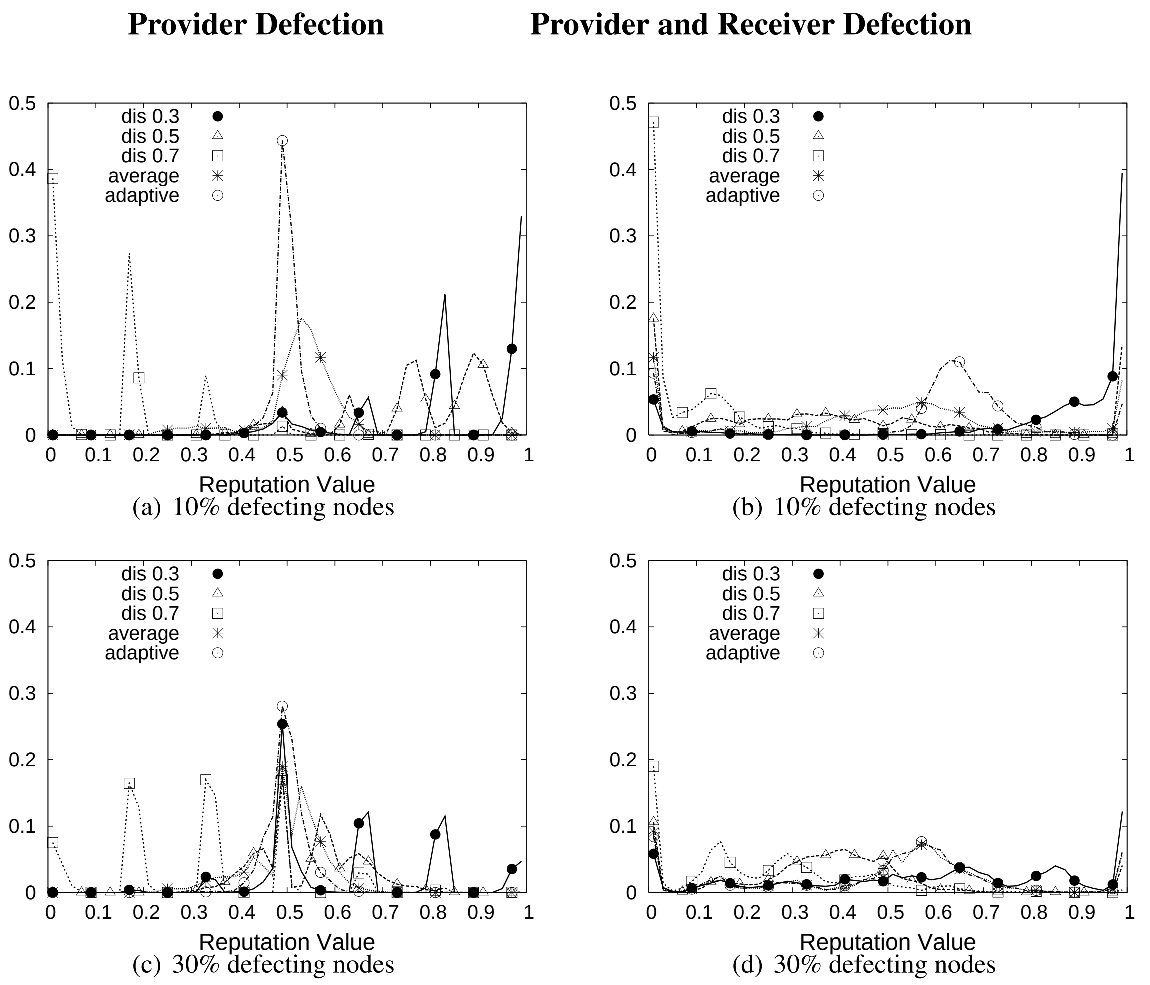


Figure 5.11: Probability Distribution Function (PDF) of the reputation values of the nodes in the system with respect to different strategies.

value of the nodes. On the contrary, nodes that choose a threshold of 0.7 do not want to risk their transactions and prefer to defect if they are not sure about the outcome.

Chapter 6

Application of reputation management schemes to content distribution

6.1 Introduction

End-system cooperation is the latest step in the evolution of content distribution on the Internet that has progressed from mirrors to proxy caches to server farms to Content Distribution Networks (CDNs). Collaborative end-user content-distribution systems are fast becoming the new paradigm for content delivery as they are cost-effective, inherently scalable and more resilient to network and equipment failures.

These systems are useful when several users are interested in downloading the same file from a single server. They exploit the fact that while server bandwidth is limited, idle bandwidth is available between the users themselves. The target file can be split into blocks which can be downloaded simulatenously by different end-users. The end-users can then collaborate and share these blocks with each other to reconstruct the original file [32, 54]. This reduces the load on the server and makes more efficient use of available bandwidth, reducing the overall download time [15].

Moreover, as more nodes join the network, more bandwidth becomes available between end-users making the system inherently scalable. Autonomic collaborative systems also successfully handle flash crowds caused by documents that suddenly become very popular putting the origin server under strain.

6.2 Misbheaviour in content distribution networks

However, making end-systems service providers in addition to being service consumers increases reliance on machines that may be less secure than dedicated servers or proxy caches that formed the backbone of earlier systems. Even worse, some of these systems may be controlled by malicious users that wish to disrupt the content-distribution network.

Since the participation of nodes in a collaborative network depends strongly on whether, and how much, they benefit from it, ensuring compliance of individual nodes to the overall goals becomes critical to the success of the system. As discussed in Section 2.3, two major concerns that have been the subject of research are free-rider, who download files without *giving back* to the network, and malicious users who introduce fake or mislabelled blocks in the network making reconstruction of the original file difficult if not impossible. In this chapter, we address the second of these concerns.

If tampered or mislabelled content is downloaded, the receiving peer must attempt to retrieve the content again. If the fake blocks can be identified, only those blocks need to be downloaded again. However, if there is no mechanism to check the integrity of individual parts of the content, as is often the case when the blocks are encoded, the entire file has to be downloaded again. This magnifies the impact of malicious behavior on the network. Therefore, the behavior of the nodes participating in the content distribution must be monitored carefully.

6.3 Approach to thwart malicious behaviour

We explore ideas of reputation management that have been developed in other contexts and apply them for soft management in autonomic and collaborative content distribution systems. We examine the ROCQ scheme, (Reputation, Opinion, Credibility and Quality), proposed in [50, 49] and described in Section 3.7, a feedback-based trust management system that uses opinions from past interactions to measure the trustworthiness of a peer. We aim at incorporating methods to sustain self-optimization of the resources by limiting corrupted downloads and to provide a minimal degree of protection of the shared content.

In this work, we extend the ROCQ model to function in a collaborative content distribution system. We simulate both scenarios we mentioned above, i.e., when the node(s) providing the fake blocks can be identified and when they cannot. We then run detailed simulations of the latter case – which we believe is a more accurate representation of systems currently in place – and measure the impact of several variables on the performance of the reputation management scheme.

6.4 System model

The ROCQ reputation management scheme is independent of the underlying cooperative content distribution mechanism. ROCQ was initially proposed in the context of virtual communities where all transactions are one-to-one. We now modify it to be used with content distribution scenarios where a single file is encoded into several blocks. We give a brief description of the how the system is modeled as details on the ROCQ reputation management system can be found in [50, 49] and in Section 3.7.

We assume that each individual peer is identified by a unique ID for the purpose of rating and this identifier cannot be tampered or changed. The global reputation values are stored in a **decentralized** fashion using multiple designated agents, or *score managers* [50, 49], for each individual peer. Before entering a transaction, a peer retrieves the reputation value of its potential partner from the score managers in order to decide if it should go ahead with the transaction.

The basic algorithm is unchanged in content distribution networks except that now a transaction fails or succeeds and opinions are formed only after **all the blocks** required for reconstructing a file have been downloaded. If the file cannot be correctly reconstructed after receiving the required number of blocks, one or more of the blocks are corrupted. If blocks can be individually checked only the senders of bad blocks will get a negative rating. If they cannot be checked all senders will receive a negative rating.

6.4.1 Content integrity

Once the content is obtained, the peer must check the integrity and the correctness of the data downloaded. This integrity checking must be accomplished to completely judge the behavior of other peers in the transaction so that affordable reports on the interaction can be filled.

Methods to assess the integrity of the content and the data contained is fundamental to prevent malicious peers to spread tampered content or, worse, virus or similar harmful content. In a content distribution network, false information may cause the peer to request the same content from other sources wasting networking bandwidth.

The integrity of a file can be calculated by hashing the file and by matching the result with the public hash value associated to that data. Unfortunately this solution requires the peer to obtain the full data before being able to check the integrity. To prevent such situation when dealing with large content, the hash value of blocks of original files may be calculated and disseminated in the network. Such solution forces a user to obtain a block unit to check the integrity. The cost in terms of bandwidth and computations resources increases as the number of blocks increases.

If the content transmitted is encoded by the source, simple cryptographic hash functions are inadequate for authenticating data. For rateless erasure code on-the-fly verification of the blocks is possible [67] if a secure separate channel exists. A similar scheme can be used in other contexts to reduce the loss of bandwidth caused by corrupted blocks to a minimum [55]. More complex schemes are based on homomorphic signature to ensure integrity of the content [29].

6.5 Experimental evaluation

In this section we study the performance of the ROCQ reputation management scheme in reducing inauthentic downloads and in detecting malicious peers in a collaborative content distribution network. We use FreePastry [85], an open-source implementation of Pastry that is written in Java, to locate and route messages over this peer to peer network.

The reputation management scheme is implemented as an application that runs on top of individual Pastry nodes. The parameters' settings and performance and decision metrics that we use to evaluate the results obtained from the simulations are listed below and summarized in Table 6.1.

Number of nodes and interactions. In our experiments, we simulate a network of **1,000** peers with **5,000** content transactions taking place in each simulation run. Note that each transaction involves downloading of k blocks, where k lies in the set $\{1, 2, 4, 8, 12\}$. The default number of score managers storing reputation ratings for each peer is 6. Each experiment is performed 10 times and the average of the results is plotted.

Performance metric. To evaluate the performance of the reputation management scheme, we calculate the proportion of successful file transfers as a

Table 6.1: Experimental parameters

Parameters that are fixed for all experiments

# nodes	$1,000$
# content transactions	$5,000$
# score managers	6
# experiments run	10
type of node maliciousness	reports and file
decision metric	reputation plus local opinion
type of decision	deterministic
trust threshold	0.5
network topology	random

Parameters that vary in the experiments

# blocks	$k \in \{1, 2, 4, 8, 12\}$
$\%$ malicious peers	$frac. \in \{1, 2, 4, 8, 12, 16\}$

proportion of the total number of content transactions. We also compute the number of correct decisions made (i.e. interactions with good peers that proceeded plus interactions with malicious peers that were avoided) as a proportion of the total number of decisions made.

Types of maliciousness. We assume that peers behave maliciously in both the content distribution system and in the reputation system. A malicious peer will send a corrupt block in response to a request. It also gives an incorrect opinion (or reputation) value to another peer in its capacity as a transaction partner (or score manager). Hence, if O (R) is the actual opinion (reputation) value, the value that is sent is $(1 - O)$ $((1 - R))$.

Decision metric. A peer decides whether to interact with another peer based on a combination of the reputation value obtained from the score managers and of the local opinion value that a peer forms based on the first-hand interactions. These two values are averaged by using eq. 3.5 defined in Section 3.4.5 and the decision is taken based on the resulting score.

We use a threshold of 0.5 and an interaction takes place only when the score exceeds this value. If a peer does not have a local opinion of the behavior of the correspondent peer (thus, they have never interacted), only the reputation value is used. When there is no reputation information available for the correspondent peer, the interaction takes place but this is counted as initial interaction.

Selection of the peers. The simulator is round-based where each round corresponds to the **complete** transfer of k blocks of a file to a peer, where k lies in the set $\{1, 2, 4, 8, 12\}$. At the start of each round we pick a peer at random (source) which then requests blocks from k peers in the network (targets).

The same peer can not be selected as a target more than once in the same round. The source checks the trust values of each target and rejects targets that have a reputation value below the threshold. A replacement target is found for each rejected target till we find k targets with sufficiently high trust values.

6.5.1 Comparison with the no reputation management case

In this experiment we compare the performance of the extended ROCQ scheme for content distribution networks with the case when no reputation management scheme is in effect. We evaluate the performance of the reputation management scheme both when it is possible to identify the fake block and when it is not possible to do so.

When it is not possible to identify the fake block(s) in a collaborative download, the content is assumed to be successfully downloaded only if all the k blocks are uncorrupted. If even one of the blocks is corrupt, the entire download has to be repeated. Since all the peers that sent a block are potentially malicious, the recipient peer files a negative report for each of these peers. On

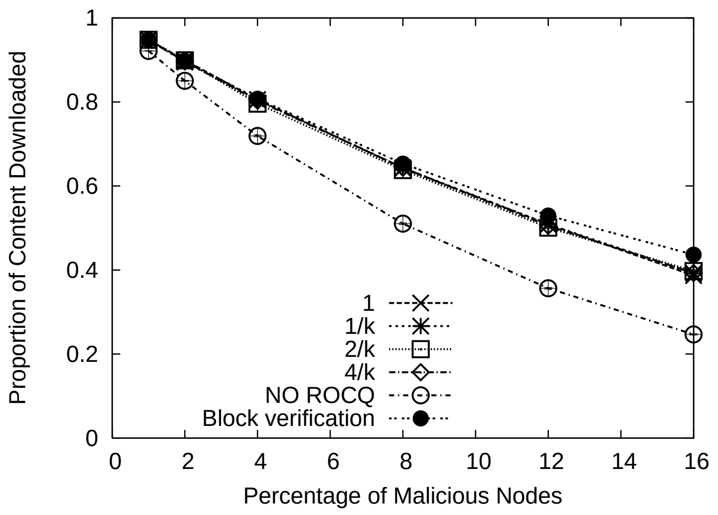


Figure 6.1: Comparison of the reputation management scheme with the case of no reputation management implemented in the system. A file consists of 8 block ($k = 8$).

the other hand, if the file is successfully reassembled, all the peers receive a positive trust rating.

Because one fake block can result in several negative reports, we also experiment with reducing the weight of negative opinions. Positive opinions are always given a weight of 1, while negative opinions are given the weight: 1, $\frac{1}{k}$, $\frac{2}{k}$ and $\frac{4}{k}$ (where k is the number of blocks and the peers involved in the transaction).

We simulate the source requesting 8 blocks (thus $k = 8$) from different peers in the network. Fig. 6.1 shows that the reputation management scheme results in up to 16% more correct downloads than when no reputation management scheme is used. We also see that varying the weight given to negative opinions does not have any significant impact on the performance of the reputation management scheme.

The reputation management scheme performs well in both cases when individual nodes uploading malicious content can be identified and when they cannot be detected. This is because in ROCQ a peer decides to interact with

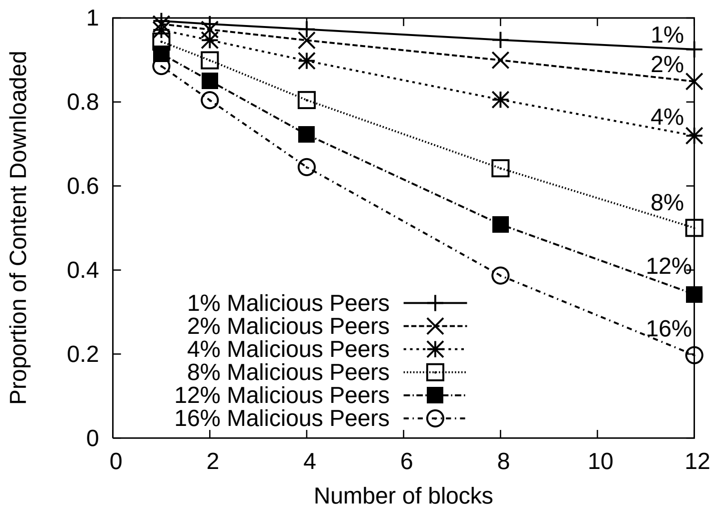


Figure 6.2: Impact of the number of required blocks on the proportion of successful downloads.

another peer based on its reputation value. As a result, malicious peers may be easily detected before initiating a transaction. Therefore, a fine-grained detection of malicious peers only slightly improves the performance of the reputation management scheme. In the remaining experiments, we assume that individual bad blocks cannot be identified as this is the worst case scenario.

6.5.2 Number of required blocks

In this experiment we measure the impact of the number of blocks required to reconstruct a file on the performance of the reputation management scheme in the distribution of content from multiple sources. Fig- 6.2 and 6.3 show that the proportion of content successfully downloaded and the percentage of correct decisions decrease as we need more blocks to reconstruct a file. We plot different lines for different percentages of malicious peers in the network (ranging from 1% to 16%).

While a larger number of blocks for the same file implies smaller block size and greater collaboration resulting in better performance, i.e., more available sources at the same time, it also increases the risk of exposure to malicious

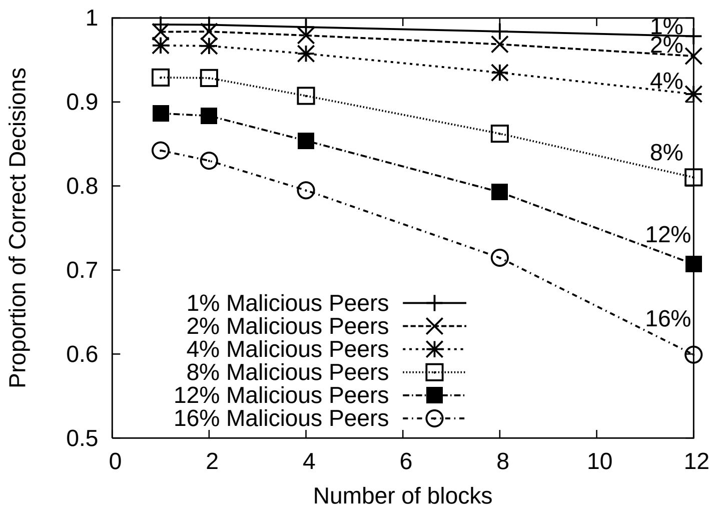


Figure 6.3: Impact of number of required blocks on proportion of correct decisions.

nodes. Our experiments do not attempt to measure this trade-off between performance and exposure to maliciousness but they do indicate that as the proportion of malicious nodes in the network increases, the optimal number of blocks that should be needed to reconstruct a file decreases.

6.5.3 Proportion of malicious nodes

In Fig. 6.4 and 6.5 we plot the performance of our scheme with different percentage of malicious nodes in the network. As the proportion of malicious nodes in the network increases, we expect the proportion of correct downloads to decrease as well. The different lines represent the different number of blocks considered during the simulation. In all the cases, the percentage of malicious nodes negatively impacts on the proportion of correct decisions made and the number of content successfully reassembled.

This is to be expected as when peers are malicious they do not provide correct blocks and the aggregation of the file can fail with higher probability. Furthermore, we are considering peers that are malicious both for the content provided and for the reports on the transactions they fill.

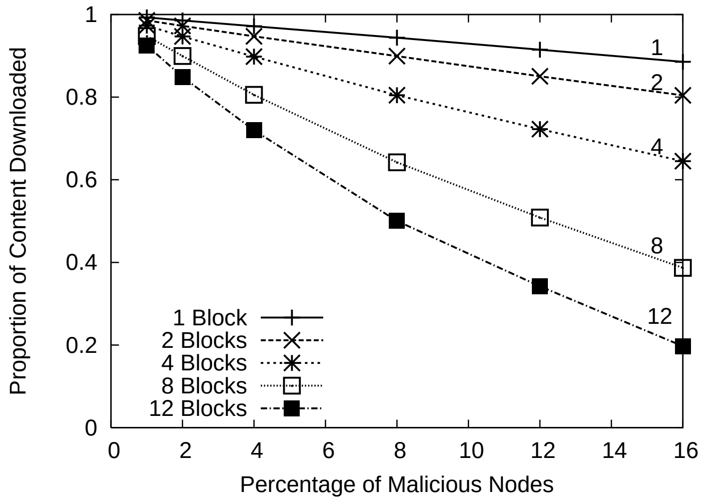


Figure 6.4: Proportion of successful downloads vs. proportion of malicious peers.

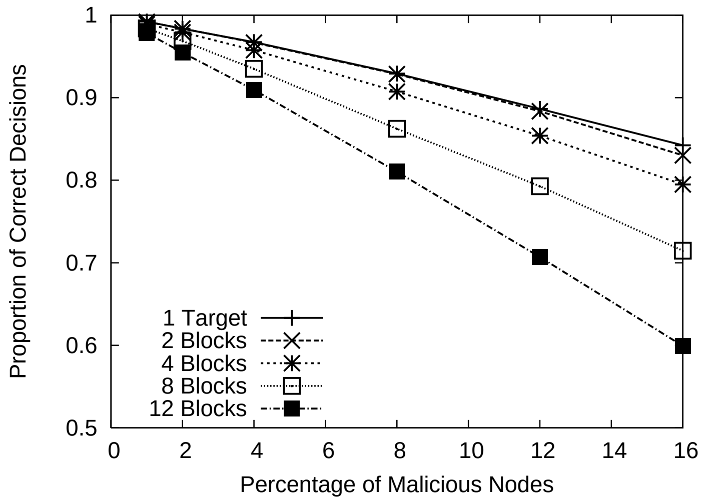


Figure 6.5: Proportion of correct decisions vs. proportion of malicious peers.

The latter impacts on the decisions made by the source when selecting the targets. The same behavior has been analyzed in [50].

6.5.4 Occasional maliciousness

When peers act maliciously in a consistent fashion, it should be possible to identify them relatively quickly. However, if peers choose to act maliciously only some of the time, it may be more difficult to identify such **occasional cheaters**.

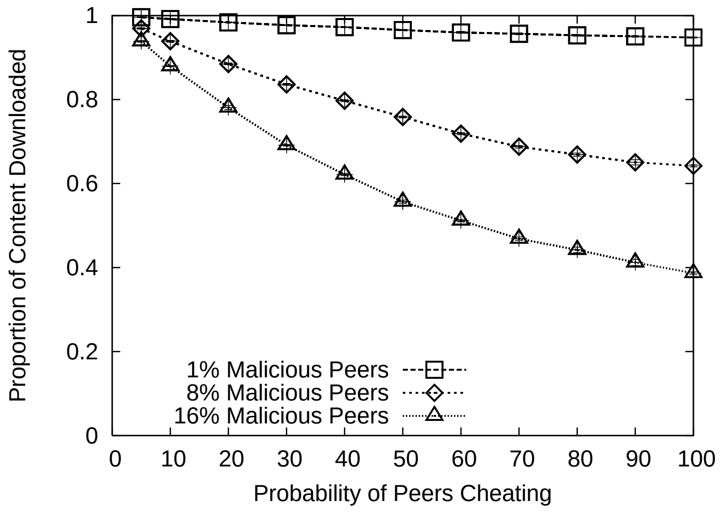


Figure 6.6: Proportion of successful downloads vs. chance of peer acting maliciously (with $k = 8$).

In Fig. 6.6 and 6.7 we examine the case when the malicious peers cheat only some of the time. We simulate the source requesting 8 blocks from different peers in the network (thus, $k = 8$). The three curves correspond to a total of 1%, 8% and 16% of the nodes being malicious.

Fig. 6.7 shows that the proportion of correct decisions is only slightly affected by the probability of a node cheating. However, Fig. 6.6 shows that proportion of successful downloads decreases much more sharply as the percentage of time nodes cheat increases. Since the proportion of correct decisions has decreased only slightly, we can see that ROCQ has identified malicious nodes with the same relative success. The reduction in the proportion of correct downloads can be attributed simply to the larger proportion of malicious transactions. This also demonstrates how in a collaborative content distribution network, a little bit of malicious behavior can have a disproportionate impact on the network.

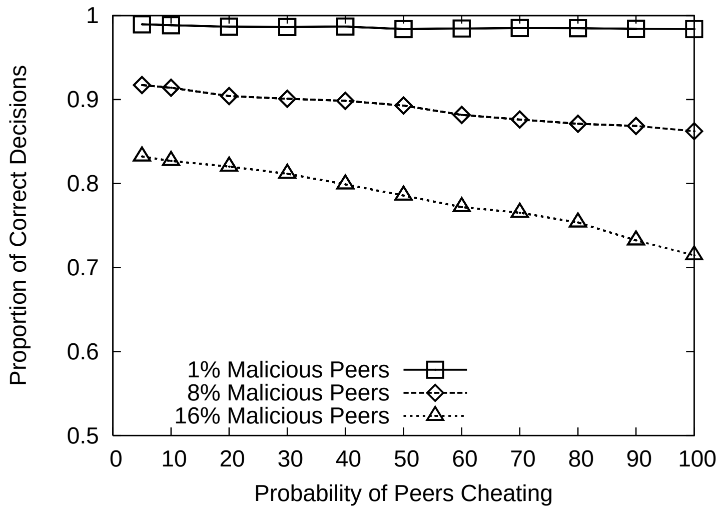


Figure 6.7: Proportion of correct decisions vs. chance of peer acting maliciously (with $k = 8$).

6.6 Discussion on content distribution networks

Several techniques for collaborative content distribution have been proposed recently. Initial proposals focused on extending overlay multicast architectures and constructing multiple parallel overlay trees such as in SplitStream [28]. Mesh architectures provide an alternative to tree-based systems allowing end-nodes to benefit from additional connections. The most popular of these is BitTorrent [32] which is a peer-to-peer application to enable fast downloading of popular files. Because there is no predetermined path in meshes and content can cycle within the mesh indefinitely, nodes need to coordinate download decisions. BitTorrent's solution is to download random blocks in the initial phase of a download and then download the block that is rarest in a node's neighborhood. While this approach is simple, it is not globally optimal.

Alternative schemes have proposed block encoding as a method to improve the efficiency of content propagation by reducing the need for node coordination. These schemes allow a large number of distinct blocks to be generated from a file. The advantage of coding is that only a small subset of the blocks has to be downloaded to reconstruct the original file. However, care must be taken to ensure that the blocks do not overlap too much. Two popular encoding approaches are *erasure codes* [23, 24] and *network coding* [9, 54]. In the former, new codes can only be generated by the server, whereas in the latter, any node can generate new codes for a file based on a linear combination of all received blocks for that file present at that node.

However, encoding blocks makes the problem of verifying block integrity more difficult. A simple checksum is no longer sufficient to check whether a block has been tampered with in transit. The problem is compounded when *rateless codes* are used, as in this case, the total number of codes that can be generated is very large. Krohn *et al.* [67] have proposed a homomorphic collision-resistant hash function as a solution to this problem. Their solution requires the source node to generate a hash value for the entire file and other nodes in the network to download this hash to be able to verify block integrity on-the-fly. They show that this homomorphic hash function is independent of the encoding rate allowing it to be used with rateless codes. Nevertheless, the problem remains unsolved in the case of network coding where any node in the network can generate a new code. Therefore, it is not possible to generate hash values for all the possible codes *a priori*.

The techniques to protect content focus on checking the integrity of individual blocks. If a block is corrupted, it can be identified and all the downloaded blocks for that file does not have to be discarded and downloaded again. Using reputation systems, we can identify malicious nodes and thus prevent downloading of the corrupted block thereby saving even more bandwidth. Moreover, the reputation approach works even in the case of network coding.

Incentive mechanisms have also been proposed to combat the problem of free-riding and to protect the content distribution network from malicious nodes. BitTorrent introduces *tit-for-tat* to combat free-riding by requiring nodes to upload content in order to continue downloading. The difference between the amount downloaded and uploaded by any node is bounded.

In [54], two incentive mechanisms are proposed to discourage free-riding for network coding: exchanges have higher priority over free uploading and a peer does not upload content if it has not received enough content from the requiring peer.

6.6.1 Improvement of reputation management systems

In this chapter we presented an application of reputation management systems and experimental results after applying an extended model of ROCQ to collaborative content distribution networks. Unlike other reputation management schemes, our approach targets collaborative content distribution by splitting files into blocks and uploading them to different peers followed by the exchange of these blocks amongst peers.

An immediate applicability of the scheme, proposed in this chapter, is to content networks relying on network coding. With network coding, blocks are re-coded at each intermediate node, and block verification cannot be done by using a predetermined hash of the block. Moreover, the scheme proposed by Krohn *et al.* in [67] is computationally expensive.

Our solution to the problem of malicious corruption of blocks is to allow a peer to select its transaction partners based on the trust he/she places on them. A collaborative content distribution system is particularly susceptible to *pollution attacks* since a single bad block can render the entire download worthless [70]. With ROCQ, a large percentage of malicious nodes are identified and corrupt block transfers are prevented.

The experimental results demonstrate the advantage of using reputation management systems in a collaborative content distribution network as opposed to not using any. We show that as the number of blocks required increases, the probability that the source downloads the content successfully decreases. Hence networks with a large number of malicious nodes should keep the number of blocks required to reconstruct a file as low as possible.

Chapter 7

Mobility support for reputation management systems

7.1 Overview

Autonomic communication networks are self-organizing systems implemented at the application layer that enable nodes to interact with no regards to which physical network they are connected to. The easy deployment of these systems has contributed to their wide diffusion and use by entities who join peer to peer systems mainly to share content. Following this trend, users must be able to profit from peer to peer systems also while they are on the move.

In this chapter we focus on the implementation of a reputation management system in a mobile environment. The mechanisms and the approach, herein defined to properly integrate mobile devices in a reputation system, can be applied to any distributed system that implements a reputation management scheme.

7.2 Introduction

The increasing use of wireless networks and portable technologies, such as laptops, cellular phones, and Personal Digital Assistants (PDAs), has enabled nodes to connect to internet whenever access is available. Nowadays, shopping centres, airports or city areas for people aggregation and entertainment are widely covered by hot spots. Nodes can connect easily either after joining a community or after subscribing to operator services.

This new communication paradigm has changed the role of the user in the network and the way self-organizing systems, like peer to peer networks, are used. Furthermore, device mobility and user mobility create new interesting services, such as food recommendation or news sharing, designed specifically for nomadic users.

The integration of peer to peer technology with mobile applications brings new interesting opportunities both for mobile consumers and wireless providers. This integration has kept the attention of researchers to provide new service platforms able to integrate the two technologies [64]. This new paradigm also enables the coexistence of heterogeneous devices, which are the basic components of any autonomic system. However, the need for trust relationship and for mechanisms to foster cooperation between mobile entities remains the same as in fixed networks, if not of greater importance.

In this scenario, the survivability of self-organizing systems relies on mobile entities who must contribute in terms of bandwidth, storage, battery and services. But, as presented in previous chapters, the nature of the nodes is not prone to follow instruction toward the social welfare. Nodes tend to be selfish, i.e. they look after their own interest and do not share resources with the aim to increase their utility, refer to Chapter 5, or in the worst case malicious, i.e. they inject false content or misbehave just for the sake of disrupting the network functionality, as we have discussed in Chapter 6.

The problems caused by these two types of entities have different effects on the system performance, but the need to reduce the risks of possible *attacks* of selfish and malicious entities is the same, as defined in Chapter 2. However, the fact that nodes are on the move and they can change location and join different virtual communities introduces new problems to give an accurate estimation of nodes' reputation.

In this chapter, we define a new mechanism based on the use of a personal token so that the transactional history of a node in different locations can be correlated. Nodes are not considered anymore as new entrants in the systems and they have an initial reputation value useful to gather *privileges*. The application of this mechanism targets mobile nodes, but it can be applied to scenarios where a node is interested in different communities or contexts.

7.2.1 Introducing reputation in mobility context

Mobility is an issue in distributed and self-organized systems as user relocation results in a high churn rate. Approaches similar to tit-for-tat as in BitTorrent [32] or exchange-based mechanisms [12] that implement a symmetric and simultaneous service provision are ineffective because nodes change frequently location, i.e., community. Traditional distributed mechanisms for peer to peer systems rely on designated agents to store reputation values in a community, but high churn rates can cause instability in the systems as new nodes join the virtual community very often, see Chapter 3.

To implement a symmetric scheme or to render effective the use of designated agents for storing reputation values, nodes can form only one big community organized in a hierarchical structure [71] with at the lower layer the groups that nodes join while on the move. However, the feasibility of this solution is limited by network boundaries and huge delays in transferring information between nodes. In addition, in nomadic applications and self-organizing systems there is the need to exploit locality both for the information and service provision.

Another solution is to make the community ask other clusters for the reputation value of nodes that have just entered. This option is rather expensive as nodes move frequently and it is difficult to trace back the relevant history of a node.

If we do not count the previous history of a node or do not correlate the information among communities, nodes are considered *strangers* and rated as new entrant when they join a new community. These nodes can hardly start to benefit from resource provision as other nodes do not initiate transaction with them because they are unknown and might be reputed to be malicious. Systems like BitTorrent [69] give the newcomers the possibility to join a torrent by opening random connections to peers so that these newcomers can have content to distribute and to benefit from. However, aggressive or selfish implementation of BitTorrent can exploit this functionality to gather files without contributing bandwidth [79].

Other solutions consist of newcomers who query the system to ask its members to lend part of their reputation [52]. This mechanism enables new nodes to participate actively in the system after joining the community. Although this solution is appealing, it falls short to address mobile nodes that move across communities. Nodes are mainly *strangers* and the interactions are short in time to establish a *trust* relationship between nodes.

7.2.2 Approach

In this chapter, we define a token-based mechanism to enable nodes to *maintain* the history of past transactions. This solution moves the burden of storing personal information to the nodes as it is their interest to trace and maintain their reputation. The token is used to correlate the reputation values earned in different communities and it gives a first view of the node's willingness to cooperate when joining a new community.

The use of the token has a twofold meaning: 1) it eliminates the problem of the bootstrap of reputation in a new community and 2) a node can exploit its reputation to immediately benefit services in a new community. Moreover, it is interest of a community to admit new nodes, which might bring new content, and have a preliminary estimation of how this node will behave in the community. Nodes should be rewarded if they prove to be trustworthy. On the contrary, malicious and selfish nodes should be punished as they reduce the usefulness of the community and disrupt the system.

A potential criticism to the token-based approach consists of the possibility for the nodes to fake their old reports to hide the fact they behave maliciously or they can simply sell/lend their tokens to other nodes. If malicious nodes are seen as trustworthy can cause high damage to the system. We tackle these problems by imposing that the score for a node, issued by a community, is bound to the identity of the nomadic node and it is digitally signed by the community. Each message is released at periodic intervals and, in new clusters, the aggregated reputation value must reflect the time of the report to age reputation in accordance. If a record is not present in the token or the node does not disclose part of the token, designated agents consider the missed information as the node misbehaved during that period.

7.3 Mechanisms to establish trust relationships

In this section we describe the system model, the mechanism and all the operations required to transfer reputation among clusters.

7.3.1 System Model

The autonomic communication system is composed by entities that dynamically change position in an open environment. The movement can be driven by a task or can be random in the area. We envision hot spots in different locations that offer connectivity to users in a area while they are in the range of communication or ad-hoc networks formed spontaneously to enable content and service exchange. The area is delimited and it is divided in clusters, which can represent the presence of hot spots or the ad-hoc community.

We suppose that users are equipped with devices with one interface and, while on the move, nodes are associated to one cluster and they join the virtual community defined by the cluster itself. We assume that there are no central authorities that monitor the activities of the nodes and in each cluster there is one ad-hoc formed peer to peer network which is fully self-organized.

Nodes can exchange data only with other peers that are located in the same cluster. This assumption limits the communication of the user from a networking point of view, but it is in line with the definition of self-organized and ad hoc communities where nodes must exploit local resources: the information has higher value when it is contextualized to the current location of the node. Extension to our model can be easily formalized by defining a broader scope for our approach. For instance, nodes can communicate across cluster by using specific gateways that hide the location of the nodes and help for the correct function of the reputation management system.

7.3.2 Node identity and structure of the scheme

Each node is associated with an identifier. This identifier is globally unique in the system and it is used to identify the node for reputation management purposes. The reputation is formed by considering past transactions of the nodes with other peers and each node maintains a local opinion of the trustworthiness of the peers with whom it interacts.

In our model we assume that the autonomic peer to peer network is organized in a Distributed Hash Table (DHT) that is used for routing of the messages even if the reputation management scheme can function either in unstructured and structured overlay networks. Nodes' identifiers are hashed and the result gives the position of the node in the DHT. The job of maintaining the reputation score of the nodes is assigned to the node responsible for a specific portion of the table. This node is named designated agent or score manager, see Chapter 3 for definitions. The use of a distributed hash table is not required in a small network as the cost of updating the local routing table does not cover the benefit of the lookup process. However, we formulate the model by using the DHT to simplify the nomination of designated agents, as the overlay topology of the community does not impact on the performance of the token-based mechanism extension compared to the normal functionality of the reputation management scheme.

Nodes leave and join the cluster due to mobility, and, in case a score manager abandons the system, it does not transfer the reputation scores, so far computed, to the node now in charge of controlling the specific portion of the table. Thus, for resiliency, multiple score managers are used to store the reputation value of a node. This choice is also important to avoid that a single node is responsible for the reputation value of other nodes, in particular when this score manager is found to be malicious.

When a node joins the system, its initial reputation value is null (0), which corresponds to the value of an uncooperative node. Because the interactions are based on the reputation value of the requesting peer, the new entrant must initially cooperate to earn reputation in the community. This behaviour could be not acceptable by honest nodes, in particular when the system is populated by free-riders.

For this reason, there is a need to reduce the number of times a node is considered as *new entrant* in the system or to define a mechanism for the bootstrap of the reputation. As discussed above, [52] proposes a mechanism for lending reputation to new entrants which does not have applicability in mobile context as nodes are strangers and the churn rate is high. For this reason, the mobility mechanism must be designed to support transfer of reputation across communities to reduce the number of newcomers.

7.3.3 Adversarial model

We consider a system populated by only two types of nodes, as defined in Chapters 3 and 2: selfish or rational nodes and malicious nodes. Selfish nodes strategize their actions and contribute to the system with few resources to maximize their profit. Malicious nodes can inject false content in the system or they can misreport information. We specifically do not deal with collusion attacks and DoS attacks, in the sense that nodes can send multiple requests or multiple reports to overload the serving capabilities of the nodes.

The goal of this scheme is to thwart malicious behaviour and reduce the risk of impersonation, whitewasher, bad mouthing and repudiation attacks, defined in Section 2.3.2. Selfish nodes might be uncooperative for resource provision, e.g., they deny access to a service. But if they decide to join a transaction, they are assumed to follow the protocol and to be honest when reporting feedbacks on a transaction. Malicious nodes misbehave in a transaction and in reporting feedbacks. These nodes can also try to subvert the system and create fake tokens or modify the entries. For this reason the reports must be secured.

We assume that nodes obtain a certificate from a trusted authority at the bootstrap and they use their private key to sign each report. To ease the procedure of constructing certificate-based trust relationships, trust chain mechanisms can be used to set-up security associations in a complete self-organized system, as described in [92].

7.3.4 Reputation mechanism

The objective is to identify malfunctioning nodes and free-riders, and to avoid that malicious nodes access available services. To reach the goal, the ROCQ [49] reputation management scheme is used and extended. Before interacting with a peer, a node retrieves the peer reputation value and, then, it decides to provide the service to the requesting peer if this peer is willing to cooperate also in the future, i.e. it has a high reputation value. After each transaction, the two interacting nodes judge the correspondent party and send their opinion to the designated score managers.

The reputation value is calculated by averaging the reported opinion, the credibility of the reporting node and the confidence that the reporting node has in its opinion. The credibility is a measure to detect nodes that report incorrect opinions, thus, their opinions will have little weight in the aggregate functions. The quality is used by nodes to weight their opinions and to indicate how their opinions should count in the reputation calculation.

Multiple score managers aggregate and store the reputation of a node and the scores that the designated agents report before node's interactions are aggregated locally, as for eq. 3.8 defined in Section 3.7. The details of the ROCQ reputation scheme are not explained in this section and the reader can refer to Section 3.7 for a complete description of the parameters and the aggregation functions.

7.4 Mobility support: the trust token

Nodes periodically receive a report on how they have behaved in a cluster. This report is sent by the clusterheads and it is added to the *trust token*. This token is personal and it stores the reputation values associated to the activity of a node, as shown in Fig. 7.1. The token consists of n-entries to contain the history of the nodes in different periods of time. This is important to judge over more samples node's behaviour and to detect nodes that have inconsistent behaviour. Each entry contains the Identifier of the node (Id), the Identifier of the cluster (c-Id), a timestamp (T) that indicates when the report has been issued, the reputation value (R) of the node and the quality value (Q) associated to the reported score. The report is digitally signed (Sig) by the clusterheads to avoid forgery of the message and fake reports.

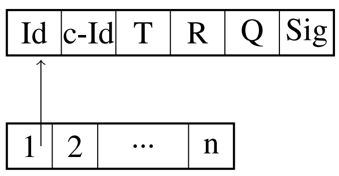


Figure 7.1: The Trust-Token is formed by *n*-entries and each entry contains: 1) Identifier (Id) of the node, 2) Cluster Identifier (c-Id), 3) Timestamp (T) that indicates when the reputation value has been issued, 4) Reputation (R) value of the node, 5) Quality (Q) value associated to the entry, 6) Cluster digital signature (Sig) on the report.

The node identifier (Id) is used to bind a report to the node. This binding ensures that the node does not transfer the token to other entities or that the node uses the same token to enter a new community with multiple identities.

For instance, lets suppose that a node has built a high reputation value and it creates several identities, it could decide to join a new community with either a single identity or multiple identities and behave maliciously. In the former case, the malicious node is not punished for his behaviour and nodes are incentivized to misbehave inside the system. In the latter case, the actions of the malicious peer can have high impact on the performance of the system, as it controls multiple nodes that have a high reputation and due to this privilege, the malicious node can cause more damages in the system.

The cluster Identifier(c-Id) specifies the issuer of the token. This has a twofold meaning: 1) other clusterheads can use the correct cryptographic material to verify the signature of a report and 2) nodes inside a cluster can determine the capability of other clusters to *recommend* nodes, i.e., the cluster credibility. In our extended version of reputation management scheme to support mobility, the confidence that a node has in the reporting capability of a cluster is defined as *cluster credibility*. This credibility factor is used to weigh the reporting values an entrant node brings inside a new cluster to form the initial reputation value.

The behaviour of nodes entering a new community will impact on the credibility of the cluster that issues reports for the nodes. If a cluster gives wrong reports about peers, its credibility rating is decreased and its subsequent reports have a reduced impact on the reputation of an entrant peer. Similarly, if a cluster's report is consistently good, i.e., in agreement with the behaviour of the nodes inside the new cluster, its credibility rating goes up.

Cluster credibility ratings are based on first-hand experience only and they are not shared with other peers. The function to update a cluster credibility is similar to the function a node uses to update the confidence of a reporting node for ROCQ, eq. 3.9 in Section 3.7.

In the system, we assume the clusterheads to be trustworthy in providing the reputation values as they are the nodes that initiated the community and mainly stable inside the cluster area. They have interested in reporting correctly node's reputation; otherwise the cluster will be rated negatively by nodes that belong to other clusters. However, the clusterheads when acting as score managers compete with other nodes to access resources. They can be malicious and misreport reputation values like other agents.

The timestamp (T) specifies when the report has been issued. The timestamp is required to order the reports and to age the information contained in the reports when the new entrant reputation value is calculated. Clusterheads release reports at regular intervals and if a report is not present, null reputation value is associated to each missed report. Thus, malicious nodes have no incentives to hide negative reports.

However, if honest nodes disconnect from the system due to networking problems, they are rated as misbehaving nodes. This might be in contrast with the nature of the nodes, but they must be incentivized to be active in the system. Otherwise, from the system designer point of view, their behaviour is similar to free-riders. Hence, they must be considered uncooperative for the idle period.

This mechanism also dissuades users from not joining a community if they do not have interest in the available content. Finally, if a node is cooperative, it re-creates its reputation value in few interactions.

The reputation value (R) gives an estimation of the behaviour of the node inside a cluster. It is updated after each transaction the peer participates in. The reported score is the global trust value the node has at the time when the token-entry is released.

A quality value (Q) is associated to the reputation. It represents the confidence that the clusterheads have in their reports. Giving incorrect reports can decrease the credibility of a cluster. So, clusterheads can lower the quality value for reported reputations if they are not sure, therefore risking less loss of credibility in case their judgment is incorrect: this happens if there few samples to estimate the reputation value of a node or the node behaves inconsistently in the community.

The quality value is defined as the confidence level that the actual mean calculated for the reputation of a peer lies within the confidence interval. The equations for the estimation of the quality value can be found in [49, 50] and more details of the quality value are discussed in Section 3.7.

The digital signature is done on the hash of the report. The clusterheads sign the report to provide integrity and a proof of participation of the node to the system. The signature is verified by the members of other clusters before a report is considered to be valid. This procedure is required to avoid fake reports from the nodes and possible modification of the message. The report is not encrypted for two main reasons: the report is disclosed to different clusters and the node can track its reputation value.

7.4.1 Operations to join a cluster

While on the move, nodes change position and they can join a new community. To enter the community, they submit the trust token so that they can immediately start to benefit from the services available in the new peer to peer network. The trust token is defined as the ticket required to enter an already formed cluster.

The token is routed following the protocol communication specification used inside a community and sent to the designated agents elected for the node. For instance, the community could be organized in a DHT or be fully unstructured. It is important to notice that a node can erase the entries contained inside the token if it does not want to disclose them, but these deleted entries count as negative transactions in the computation of the reputation value. More important is the fact that the node cannot generate fake reports or modify an entry as each entry is digitally signed by the clusterheads. Thus, the public key associated to a cluster must be available in the system. Certificate issued by a common certification authority and trust chains can serve easily for this purpose.

The role of the designated agents of a new entrant is to verify the integrity and the signatures of the token. Then, designated agents compute the new reputation value which is stored locally for future use inside the community. These agents form the *first* reputation value by weighting the information the token contains with the credibility of the clusters that have issued the entries. The credibility of the cluster is not shared inside a community because it would introduce extra signalling and nodes can also lie for the credibility of the cluster. If a designated agent has no previous information on the cluster credibility, it initially considers a value of 0.5 when 0 means no confidence and 1 the designated agent is fully confident in the reporting cluster.

Table 7.1: Size (in bytes) of the fields for one token entry.

Field name	Id	c-Id	T	R	Q	Sig
Size	4	4	4	4	4	128

7.5 Communication overhead

In this section we evaluate the communication overhead of the token-based mechanism.

Clients must store the trust token that has n entries. The size of each entry is 148 bytes and the total storage for the token is equal to $n * 128$ bytes, see Table 7.1. The client and the node identifier are encoded on 4 bytes, assuming the IP-address is the Id of the client. In a small area, 2 bytes may be enough to encode the cluster identifier – 1 byte to identify the area and 1 byte to identify the cluster within the area. As defined in Chapter 4, the timestamp and the reputation values are encoded on 4 bytes. The quality value is encoded in accordance with the reputation value.

The digital signature results in 128 bytes and has great impact on the size of the entry. For our calculation, we consider RSA of modulo 1024 bits to generate the signature, but more efficient schemes based on elliptic curves can be used to reduce the size of the signature.

The number of messages depends on the frequency with which clusterheads disseminate the reports and on the mobility patterns of the nodes. Reports are distributed at regular intervals and the frequency can also be adjusted based on the persistence of the users in the system adaptively. Another strategy that can be used to reduce the overhead is to make the node ask the token before leaving the group. This solution reduces drastically the number of messages transmitted at the cost of losing an entry in case of abruptly disconnection from the cluster. Similar results described in Chapter 4 can be applied to understand the overhead of the token based mechanism.

7.6 Security of the token-based mechanism

In this section, we analyze the security solutions that can be adopted to mitigate the impact of impersonation and bad mouthing attacks, discussed in Section 7.3.3. The token-based scheme makes use of digital signatures to ensure integrity and correctness of the reports. The signature must be recognized to belong to the cluster otherwise the report is not valid. A number of options are available to enforce this method and we analyze each of these possible solutions to derive conclusions on their feasibility in the context of the token-based scheme.

We assume the existence of an off-line certification authority (CA) that issues certificates for the nodes. Thus, a pair of keys is associated to each node. This is required by the reputation management scheme as opinions must be digitally signed by the issuer after each transaction. The same procedure could be adopted by the clusterheads to digitally sign the reports.

This last setting has two main drawbacks: 1) the leaders in a cluster are many and they do not share the same key pair; 2) the verifiers must know the public key of the signer. The latter means that each mobile node should store the public keys of all possible signers present in the system. Even if this solution requires a great amount of storage for the keys, mobile devices are now equipped with enough space to store thousand of keys. A typical public key has a size that ranges from 512 bits to 2048 bits if no elliptic cryptography is used, thus, in the worst case (2048 bits) 1 MB is sufficient to store 4096 keys.

However, this solution might not be applicable in a dynamic environment due to high number of messages required to obtain valid certificates. These certificates might also be piggybacked to the transferred data or asked directly to the nodes. This is not acceptable as the communities cannot be connected and the clusterheads cannot be queried for a certificate.

PGP like schemes can be also used to establish security associations in a mobile self-organized system, as proposed in [92]. Along with the distribution of the keys, the main drawback of this solution consists of the form of the signature. If clusterheads use their own private key to sign the report, this report is associated to the identifier of the signer and not to the cluster identifier, i.e., what we want to achieve. In this setting, malicious nodes can collude and generate valid false reports unless the verifier has the compete list of authorized clusterheads. This procedure might be cumbersome as the clusterheads can change over time and the list must be updated and issued by a trusted third party to be valid. Moreover, we must guarantee anonymity to clusterheads when they sign a report to avoid possible reprisals from the node.

7.6.1 Ring signature

Ring signature is a useful cryptographic tool to provide anonymity of the signer [84]. A clusterhead, responsible for the report, creates an ad hoc ring signature composed by other cluster entities **without** their approval or their aid. In our context, the applicability of this mechanism is to preserve anonymity of the signer.

The signature is composed as follows: the signer selects a number of clusterheads, which are named ring members, and constructs the ring signature of a report r by using the public keys of the members and its own private and public keys. The report signature is verified by using a verification function that requires as input the signature, the report and the public keys of the ring members. In this scheme, the signer must be part of the ring members and the verifier knows only that one of these members is the real signer. Thus, the size of the ring must be greater than 2, otherwise anonymity cannot be guaranteed. Other schemes that extend the ring signature mechanism are Id-based ring signature [31, 48] and threshold ring signature [18, 93].

The main drawback of ring signatures is that there is no control on the formation of the ad hoc group. If the verifier does not have any knowledge of the authorized members of a group, it cannot assume the signature being generated on behalf of the group. A simple solution to overcome this problem consists in distributing the list of members authorized to sign a message.

To create a strong relationship between the signer and the cluster, we can use *Id-based cryptography* [16]. In Id-based cryptography the public key is an identifier and the private key can only be generated from the public key by a trusted authority upon authentication of the node. In our setting, the identifier is a tuple that contains the node identifier and the cluster id, e.g. *Id.clusterId*. Thus, we can ensure that a signature has been issued by a member of a cluster. In a dynamic environment, we might want to give the opportunity to the clusterheads to outsource other nodes the responsibility to sign reports. To serve our goal, we can exploit Id-based signatures schemes organized in a hierarchical structure as proposed in [96].

7.6.2 Group signature

Another approach to implement a signature on behalf of the group and to guarantee anonymity of the signer is *group signature* [30, 13]. An authority generates the private signing keys and distributes them to the members of the group. The signer only uses its private signing key to generate the signature. A verification key, common for the group, is used to validate the signature. The burden of the protocol is entirely on the authority who must generate the signing keys and the public group key.

In our case, group signatures represent the most suitable solution to ensure that only authorized nodes can generate signatures for the cluster. However, we must consider two possible scenarios due to the mobility of the nodes. as the group membership might change: 1) new clusterheads can have the privilege to issue reports and 2) previous authorized members might leave the cluster. A new group key and private keys can be generated to reflect the changes in the group. This solution is drastic and might result in low efficiency in high dynamic situations.

In [14, 66] new schemes are defined to allow nodes to join the group signatures without requiring the authority to change all the keys. [17] discusses a group signature scheme with revocation that requires to notify only the verifiers of the change in the membership. The verifier accepts only the signatures of unrevoked entities.

7.6.3 Threshold cryptography

The problem of a compromised entity is not solved by ring and group signature schemes. If the clusterhead has been compromised or the device tampered and the private key stolen, a malicious node can issue valid reports and increase its reputation value. *Threshold cryptography* [35] can be implemented to require a group of nodes to generate the signature. In a (t, n) signature scheme at least $(t + 1)$ clusterheads out of n members of the cluster must sign the same report and at most t colluding members can generate a valid signature for a different report with negligible probability.

Two schemes that implement threshold cryptography in ad-hoc mobile networks are proposed in [37] and [38]. The latter discusses a dynamic scheme that allows to increase the threshold to produce valid signature. Thus the cryptographic scheme can better react to network condition changes and to the presence of malicious entities.

Finally, we can conclude that the system designer must select the most suitable signature scheme based on the network conditions and on the type of malicious nodes in the system. If the network is congested, it is better to not implement a threshold scheme as it requires extra messages to sign a report.

7.7 Application context

In this section, we describe an application scenario that can be benefit from the use of the token-based approach to bootstrap the reputation value in a community and to interconnect the history of the nodes in different contexts.

The new business frontier in providing wireless connectivity everywhere and at any time of the day has driven the creation of wireless network communities. The service is offered by municipalities, operators and private users [4, 1, 5] in airports, train stations, shopping mall and museums [2]. Users can connect to these networks to access content or to retrieve information while they are walking around the city for shopping, leisure or business.

This scenario offers many possibilities to develop new applications for pedestrians ranging from content distribution to product advertisements. Nomadic users interact and exchange information or suggestions to enrich their knowledge or to gather multimedia content. For example, a person wants to download the podcast of the latest local news or in a shopping mall he wants to know the weather forecast to decide if it is worth to book a weekend on the sea side. These applications are few example to describe the need for content exchange and the quantity of content users transfer everyday. However, the mobile settings and the little time users have to access the resources due to mobility discourage them from downloading content. This is due to the poor quality of the service in case the information is corrupted. In Chapter 6, we show that reputation management systems improve the performance of content distribution networks as they reduce the amount of corrupted information downloaded. However, the applicability of a reputation management system requires that nodes should last for long in the community to benefit from cooperation.

In mobility scenario, we envision the presence of multiple close hot spots that *follow* the users during the day or the formation of spontaneous ad hoc communities created for the need of users to exchange information. These separated communities require the installation of different reputation management systems because hot spots might not be interconnected or they might be under the same domain. The token-based extension implements an inter-cluster solution that justifies the use of reputation management system in different virtual communities.

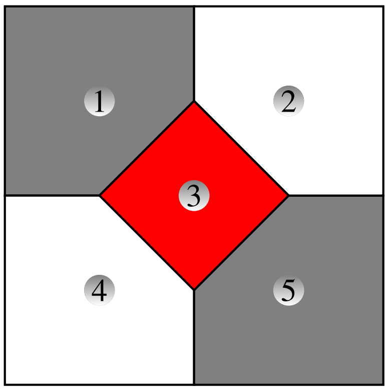
Vehicular ad-hoc networks (VANET) can also benefit from a similar approach. The limit of the applicability is on the speed of the vehicles. But during rush hours when traffic is congested in the city, the speed is relatively low and vehicles can form *quasi* stable communities similar to ad hoc networks for pedestrians.

7.8 Design and Implementation

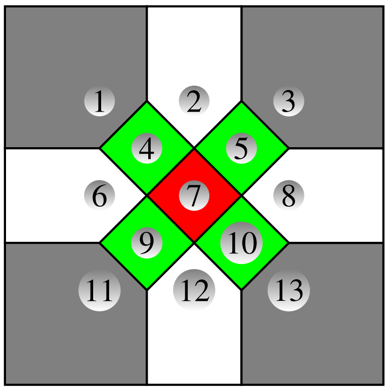
We implement a mobility scenario where nodes can move across clusters. We consider an area of $1,000$ x $1,000$ square meters, which is divided in clusters. These clusters represent the *virtual communities*. In each virtual community, a reputation management scheme is used to detect nodes' misbehaviour. We define two types of nodes: clusterheads, or more stable users, and nomadic users. The number of clusterheads is initially equal for each cluster. The nomadic users are randomly placed in the area and are divided in two classes based on the average time they remain still.

We use the Canu mobility simulator [3] to create the trace files from the mobility scenarios, which defines the movements of the nodes. The *random waypoint model* [25] is used to simulate the mobility of the nodes in the area and we extract the position of the nodes at regular intervals to define the composition of the clusters. Nodes are assigned to the closest cluster by using the minimum distance criteria between the position of the node and the center of the clusters; the center is defined by the initial position of the clusterheads.

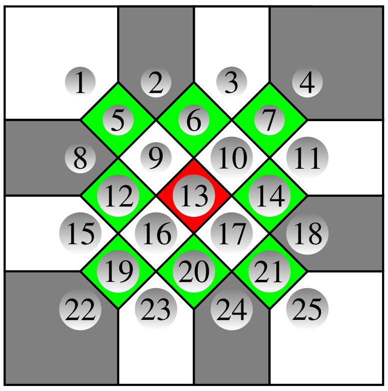
We simulate a total of 9 different settings for the same area by varying the



(a) 5 clusters



(b) 13 clusters



(c) 25 clusters

Figure 7.2: Area of 1000m x 1000m divided in clusters

number of clusters and the speed of the nodes. This results in testing the token-based mechanism in different contexts. Three different scenarios are defined based on the number of clusters. Fig. 7.2 shows the area and the position of the clusters; we divide the area in 5, 13 and 25 clusters as shown in (a), (b) and (c) respectively.

We assume that the area is open without obstacles and there are no boundaries that delimit the clusters. Thus, the movement of the nodes is assumed to be free and it follows a straight trajectory to reach the destination, which is selected randomly in accordance with the random waypoint model. When a node reaches the border area it returns in the area with the same incident angle.

Table 7.2: Parameters' setting for Slow Mobility

Class I			
Min speed	0.18 m/s	Max speed	0.46 m/s
Min stay time	30 s	Max stay time	60 s
Class II			
Min speed	0.18 m/s	Max speed	0.46 m/s
Min stay time	36 s	Max stay time	120 s
Clusterheads			
Min speed	0.018 m/s	Max speed	0.046 m/s
Min stay time	3,000 s	Max stay time	6,000 s

Simulation Time	90,000 s
Sampling period	90 s
Average number of nodes changing cluster (5)	57.9
Average number of nodes changing cluster (13)	107.1
Average number of nodes changing cluster (25)	148.2

We also define three simulation scenarios that represent the movement of slow, fast and normal peers in the area. We vary the speed of the nodes and the time a node remains in a position before moving again. For all settings, the nomadic users are divided in two classes to simulate a more heterogeneous population. The difference consist in the time they spend in a place. They are identified as class I and class II in Tables 7.2, 7.3 and 7.4.

Table 7.2 shows the parameters we use to derive the traces of the nodes' position for low mobility by using the Canu simulator. Table 7.3 and Table 7.4 show the parameters for medium and fast mobility respectively.

The combination of the type of mobility and of the number of clusters results in 9 different simulation environments to test the performances of the token-based reputation mechanism.

The simulations of the mobility models run for $90,000$ seconds. The positions of the nodes are saved in the output file every 90 seconds to have a fine-

Table 7.3: Parameters' setting for Medium Mobility

Class I			
Min speed	0.37 m/s	Max speed	0.98 m/s
Min stay time	15 s	Max stay time	30 s
Class II			
Min speed	0.37 m/s	Max speed	0.98 m/s
Min stay time	18 s	Max stay time	60 s
Clusterheads			
Min speed	0.037 m/s	Max speed	0.098 m/s
Min stay time	1,500 s	Max stay time	3,000 s

Simulation Time	90,000 s
Sampling period	90 s
Average number of nodes changing cluster (5)	113.8
Average number of nodes changing cluster (13)	207.8
Average number of nodes changing cluster (25)	279.9

grained representation of the nodes movement in the area. The position consists of the Cartesian coordinates of the node. This output file is used as input to the reputation management simulator to construct the clusters. The reputation system also requires the position of the clusterheads and the number of clusters.

We implement the token-based mechanism in Java as an extension to the ROCQ reputation management scheme. The network topology is defined by the position of the nodes and the number of clusters. In our simulator, we impose that nodes can only interact within the cluster as we want to test the performance of token-based mechanism in virtual communities that are independent among each other. The nodes in a cluster are labeled as score managers, or designated agents, only for peers in the same cluster.

Each cluster is organized in a DHT to simplify the construction of the overlay topology and the assignment of designated agents, as discussed earlier. This is not required for small networks like ours simulated one, but the use of DHT

Table 7.4: Parameters' setting for Fast Mobility

Class I			
Min speed	0.74 m/s	Max speed	1.85 m/s
Min stay time	7.5 s	Max stay time	15 s
Class II			
Min speed	0.74 m/s	Max speed	1.85 m/s
Minimum stay time	9 s	Max stay time	30 s
Clusterheads			
Min speed	0.074 m/s	Max speed	0.185 m/s
Min stay time	750 s	Max stay time	1,500 s

Simulation Time	90,000 s
Sampling period	90 s
Average number of nodes changing cluster (5)	218.7
Average number of nodes changing cluster (13)	385.8
Average number of nodes changing cluster (25)	509

does not affect the performance of the token-based mechanism. Multiple score managers are used to maintain a consistent view of the reputation values.

Due to the mobility, the cluster community changes and, thus, the score managers assigned to a peer may change over time. In Table 7.2, Table 7.3 and Table 7.4 we show the average number of nodes that move between two subsequent periods. As expected this number is function of the speed and the number of clusters. For instance, about half of the system population changes cluster when there are 25 clusters and the speed is high, see Table 7.4. On the contrary, about 6% of the nodes change community if there are only 5 clusters and the nodes move slowly, see Table 7.2.

Table 7.5: Parameters' setting for the simulation of the reputation management scheme

Number of Nodes	$1,000$
Initial clusterheads in each community	5
Number of Nodes Class I	$315 -$ # clusterheads
Number of Nodes Class II	685
Topology	random
Mobility Model	Random Waypoint
Initial trans. for reputation system	1,000
Initial trans. for reputation and token system	29,000
Number of transactions	470,000
Transactions before nodes' movement	500; 2,500
Experiments run	6
Number of designated agents	6
Type of node maliciousness	report and service
Type of decision	deterministic
Trust threshold	0.5
Size of the Token	100
Number of Clusters	5,13,25

7.9 Performance evaluation

We validate the performance of the token-based approach by comparing the results of this mechanism with the traditional application of the ROCQ reputation management scheme. In both cases, we simulate that nodes clean the stored information on the reputation and credibility values of other peers and the quality values of the nodes with whom they have interacted before changing cluster.

In our experiments we use the parameters listed in Table 7.5. There are 5 clusterheads in each cluster and the number of nomadic nodes depends on the number of clusters in each experiment. There are 6 designated agents to aggregate and store the reputation value of a node in a cluster. To calculate the trust value of a node, either the reputation value, if available, or the average opinion, if there exist a past direct transaction between the same two nodes, is used. Then, nodes use a deterministic threshold 0.5 to decide if the peers are trustworthy.

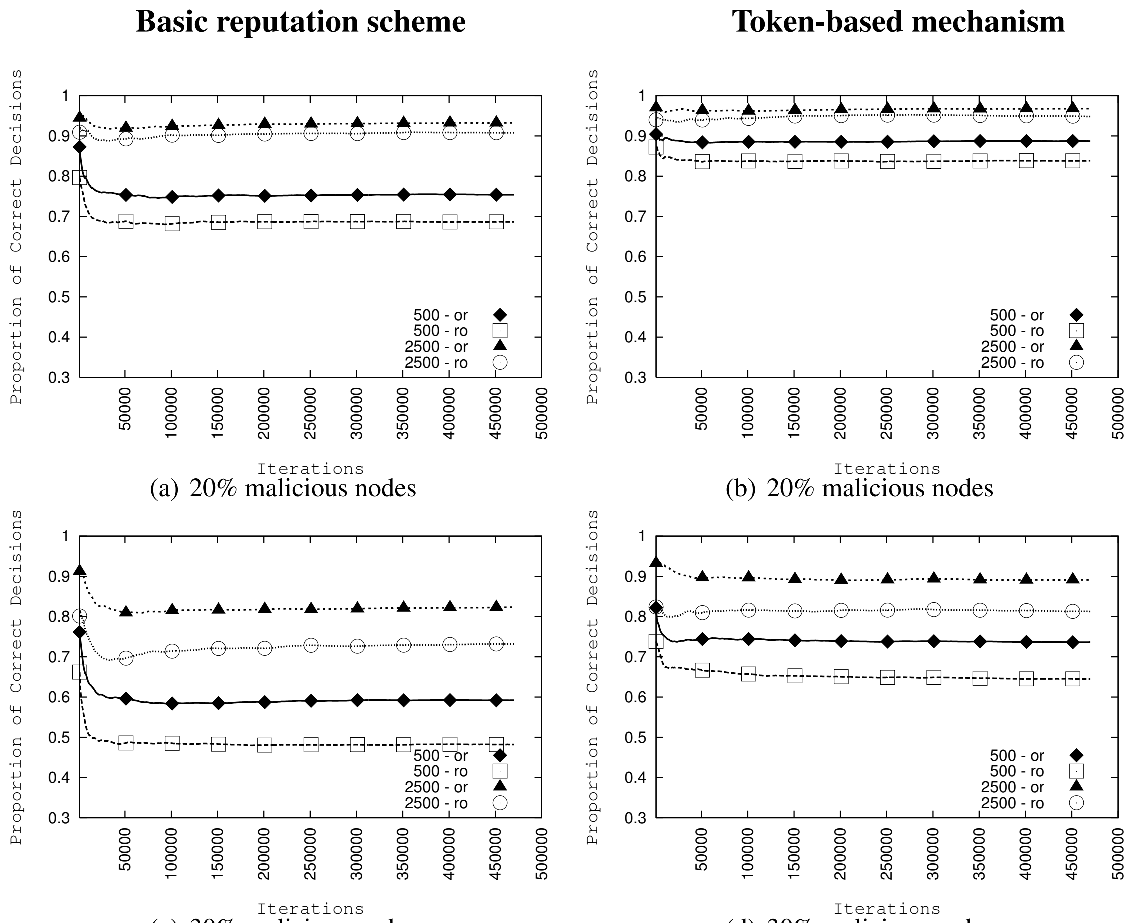
At each iteration, two nodes, that belong to the same cluster, are selected randomly to interact with each other and the result of the interaction is used to evaluate the performance of the reputation management system. We use the success rate of transaction as metric, defined by eq. 3.6 in Section 3.5.

We run an initial number of transactions to bootstrap the reputation management system and the token-based mechanism, as shown in Table 7.5. This is required to have an initial reputation value for the nodes, an initial estimation of the clusters' credibility and initial reports inside the token. The size of the token is limited to 100 records. Thus, it stores the history of the node for last 150 minutes. Reports are collected at regular intervals, with a rate that is $\frac{1}{5}$ of the frequency of mobility, which is given by the number of transactions in the system between two subsequent movements.

The membership of the clusters is updated after the movement, i.e. fix number of transactions as specified in Table 7.5, and based on the nodes' position given by the trace file.

7.9.1 Slow mobility

In Fig. 7.3 we compare the effectiveness of the token-based mechanism, in terms of fraction of correct decisions, when the nodes move slowly in the area. The characteristic of the movement are defined in Table 7.2. We simulate two metrics for the node to take decisions: reputation if there is no direct transaction with the correspondent node or opinion if there is no estimation of the reputation value, indicated by *or* and *ro* in the plots respectively. We also simulate a different number of transactions in the system between two subsequent movements, indicated by 500 and $2,500$ iterations in the plots, to study the performance of the reputation schemes when there few and many transactions to build the reputation value within a cluster.



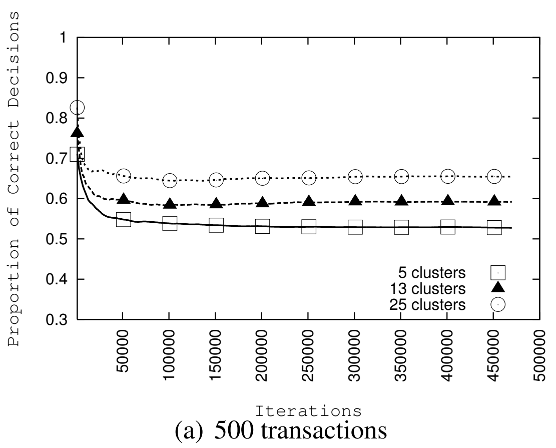
(a) 20% malicious nodes
(b) 20% malicious nodes
(c) 30% malicious nodes
(d) 30% malicious nodes

Figure 7.3: Proportion of correct decisions with slow mobility and 13 clusters.

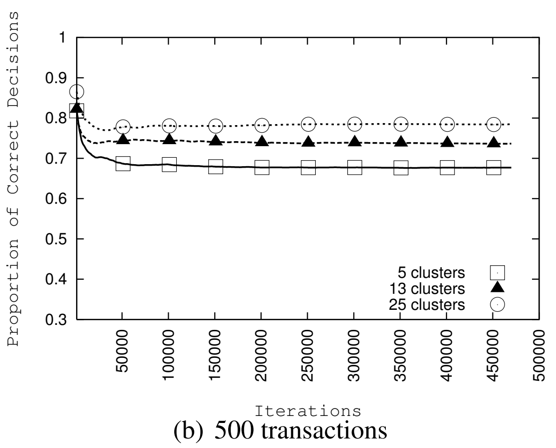
As shown in Fig. 7.3, the token-based mechanism improves the performance of the reputation management system, in particular when the fraction of malicious nodes increases in the system (plots (c) and (d)). The improvement is greater when there are few interactions in the system to form an opinion or to estimate the reputation of the nodes accurately. This is shown by the curves plotted for 500 transactions before each movement with an increase of 15% of correct decisions compared to 10% for $2,500$ transactions.

As expected, a decision based on direct experience increases the success rate in all cases. This results from the fact that the decisions of the nodes are

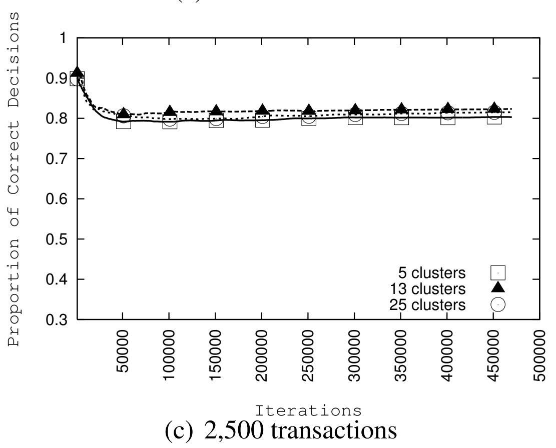
Basic reputation scheme **Token-based mechanism**



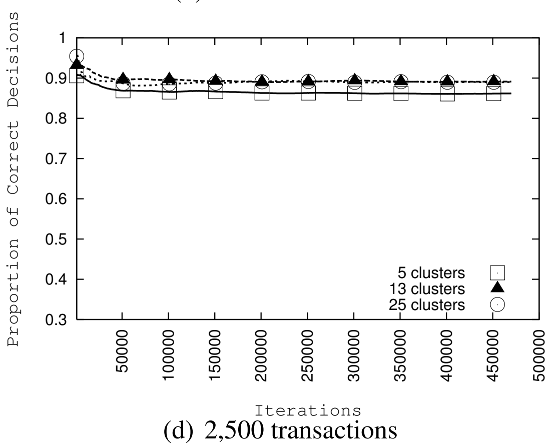
(a) 500 transactions



(b) 500 transactions



(c) 2,500 transactions



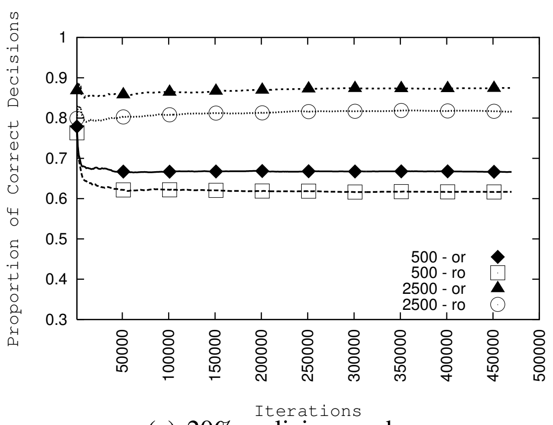
(d) 2,500 transactions

Figure 7.4: Proportion of correct decisions with slow mobility and 30% of malicious nodes.

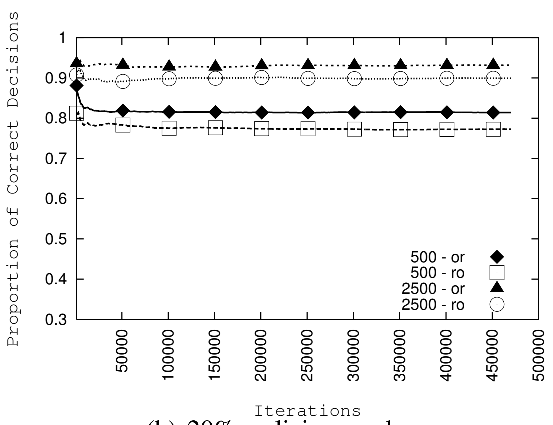
not biased by the maliciousness of the designated agents, who can report false reputation values. The token-based mechanism introduces modifications to the reputation management system for what concerns the evaluation of the reputation value. Thus, it should not increase the success rate when opinion is used first for the decisions. However, as newcomers have never interacted in a cluster after their movement, reputation, estimated with the help of the token, is used initially by nodes to take decisions on transactions.

In Fig. 7.4, we compare the performance of the reputation management systems when there is a different number of communities in the area. We plot only

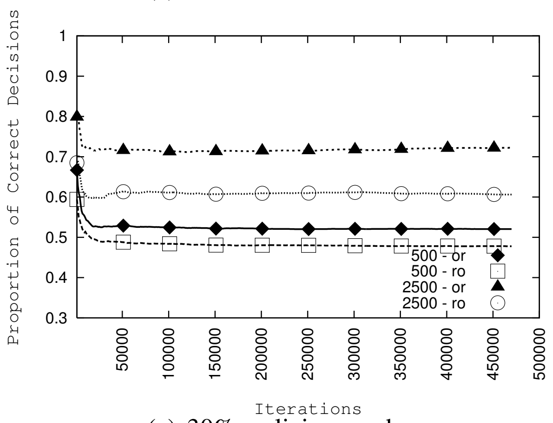
Basic reputation scheme **Token-based mechanism**



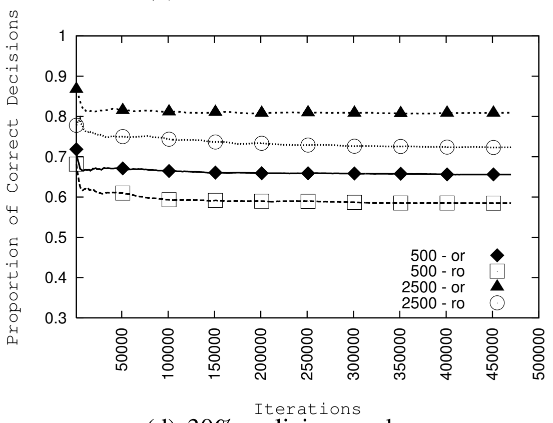
(a) 20% malicious nodes



(b) 20% malicious nodes



(c) 30% malicious nodes



(d) 30% malicious nodes

Figure 7.5: Proportion of correct decisions with medium mobility of the nodes and 13 clusters.

the case when nodes decide to transact based on direct experience as it is closer to a real scenario. When there are few interactions between two subsequent movements, the presence of more clusters gives a higher success rate, plots (a) and (b) in Fig. 7.4. This is true for both the traditional reputation scheme and the extended version. When there are 25 clusters, nodes interacts more frequently with the same nodes and as such they can form a more accurate estimation of the nodes' trustworthiness. If we increase the number of transactions to $2,500$, plots (c) and (d) in Fig. 7.4, the impact of the number of clusters is smoothed by a higher number of samples to evaluate reputation values.

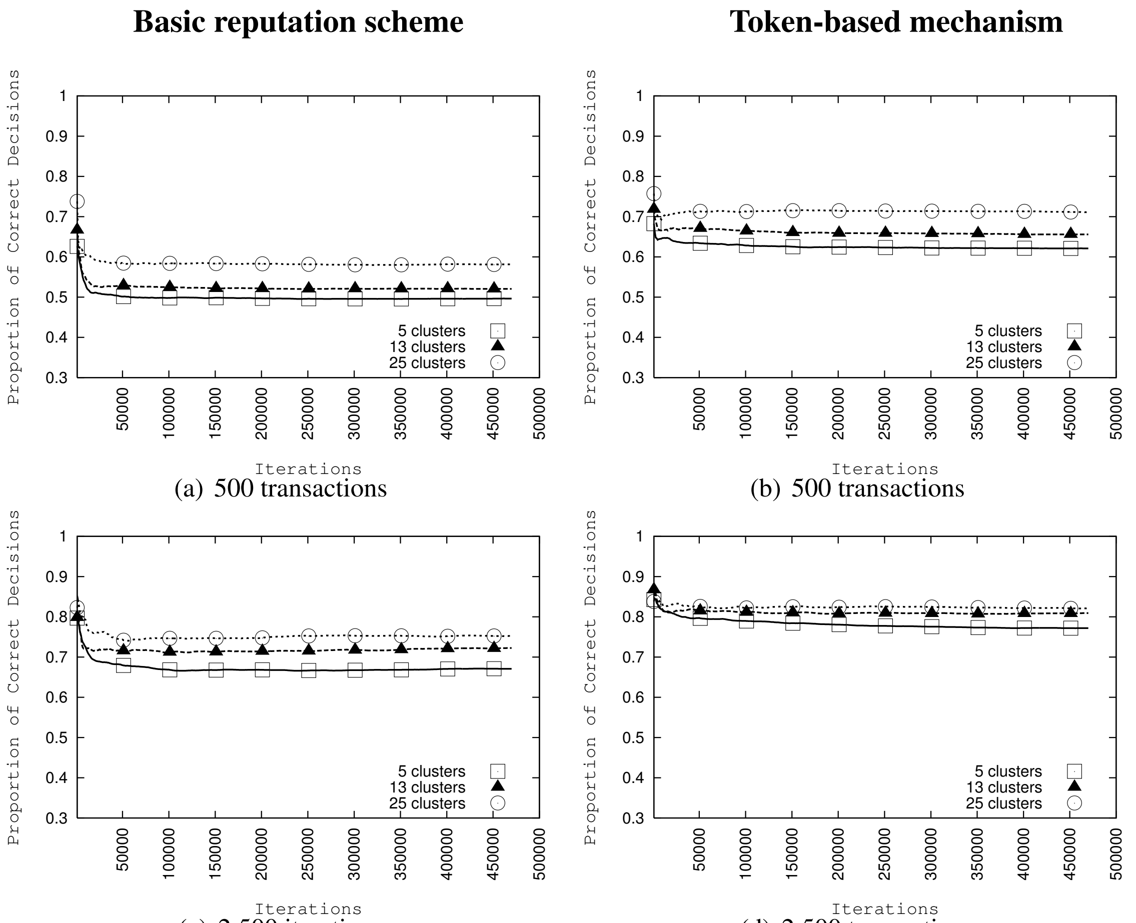


Figure 7.6: Proportion of correct decisions with medium mobility and 30% of malicious nodes.

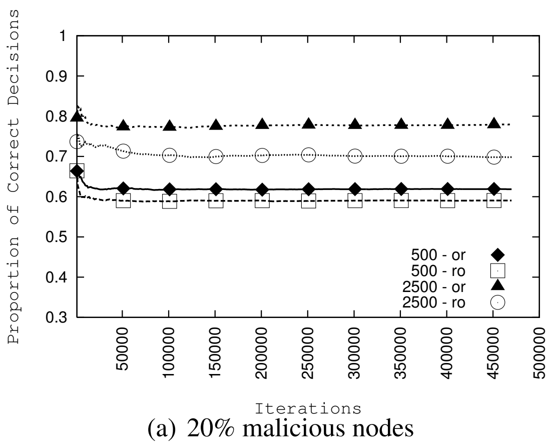
7.9.2 Medium mobility

In Fig. 7.5 we plot the success rate when the nodes move across 13 clusters with medium mobility, as defined in Table 7.3. In this case, the token-based mechanism improves the performance of the reputation management system by increasing the proportion of correct decisions. The higher mobility, compared to the slow case analyzed in the previous section, does not allow nodes to stay in a cluster for the time sufficient to have an accurate estimation of the reputation value.

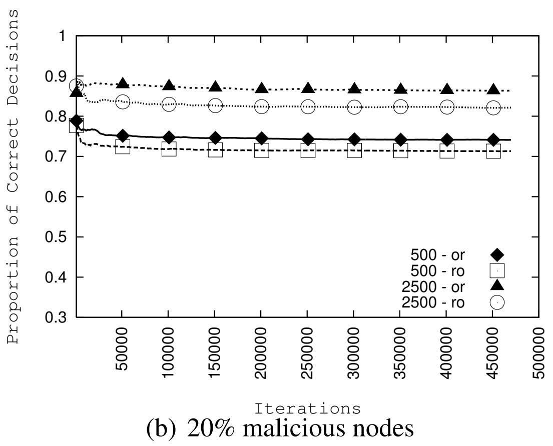
In fact, the success rate decreases by 10% compared to slow mobility when there are 500 or $2,500$ transactions in the system. This is also valid for the token based mechanism when the percentage of malicious nodes is high, plot (d) in Fig. 7.6.

But, when the percentage of malicious nodes is low, the token-based extension maintains a high precision for the reputation value even if the membership of clusters changes often. In this case the strengthness of the extension relies on the designated agents that do not send many false reports for the nodes when there are few malicious nodes. Thus, the reputation value stored in the token is more accurate.

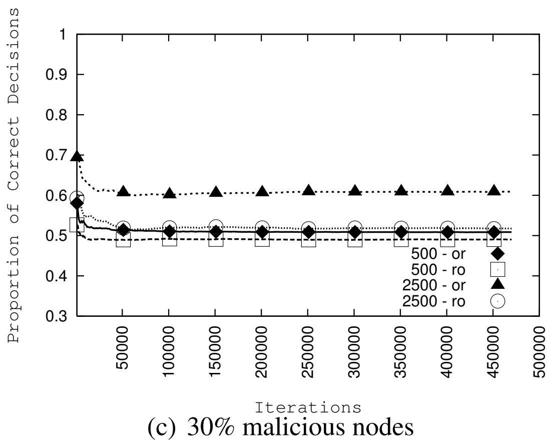
Basic reputation scheme **Token-based mechanism**



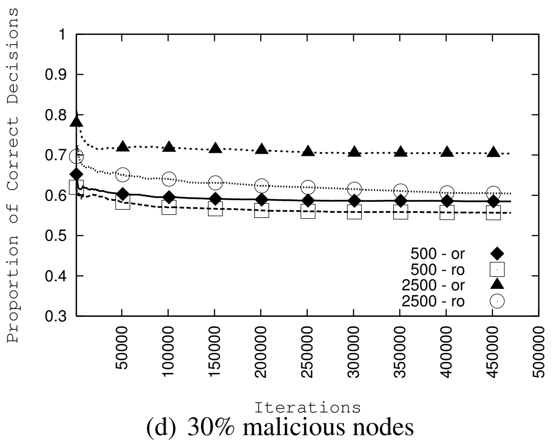
(a) 20% malicious nodes



(b) 20% malicious nodes



(c) 30% malicious nodes



(d) 30% malicious nodes

Figure 7.7: Proportion of correct decisions with fast mobility of the nodes and 13 clusters.

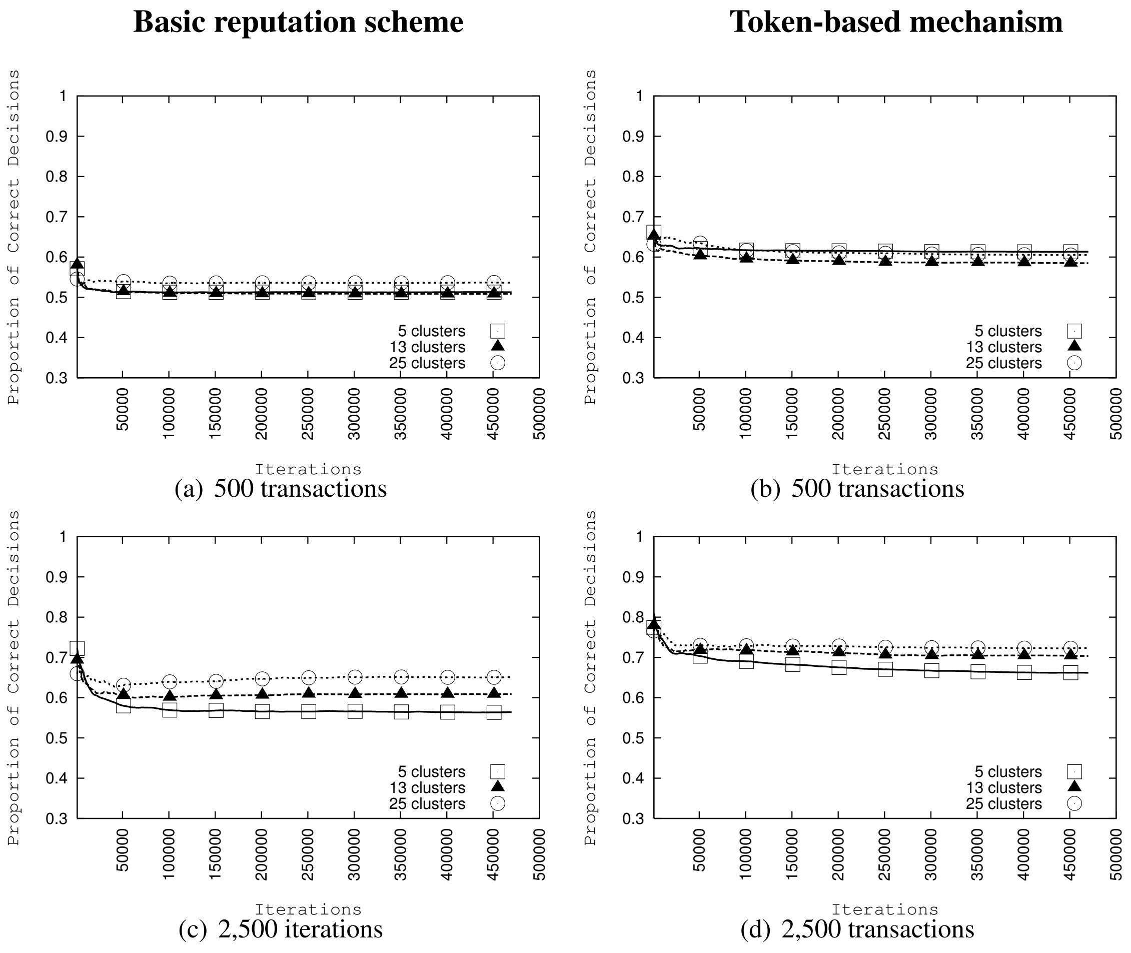


Figure 7.8: Proportion of correct decisions with fast mobility and 30% of malicious nodes.

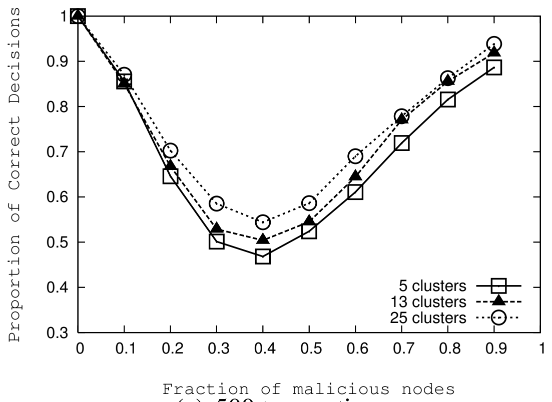
Higher mobility slightly reduces the impact of the number of clusters on the system's performance. This is shown in Fig. 7.6. When the mobility increases and there few transactions before the movement, the same pair of nodes interacts less frequently.

Moreover, when there are only 5 clusters, nodes change cluster less often, which results in a slight decreases in the performance, due to mobility, compared with the case of 13 and 25 clusters. This effect is less evident when there $2,500$ interactions between two subsequent movements, plots (c) and (d) in Fig. 7.6.

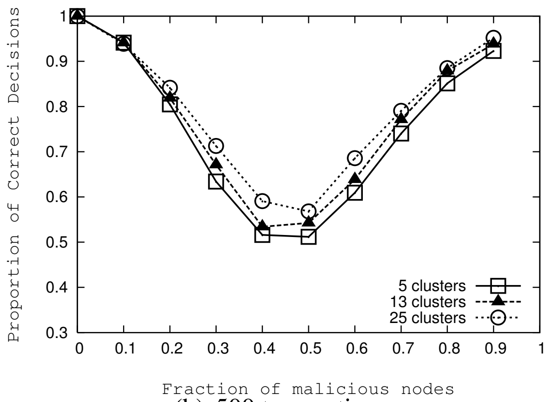
7.9.3 Fast mobility

Fig. 7.7 and Fig. 7.8 show the success rate when nodes move fast. The plots confirm the behaviour, observed earlier, of the reputation management system when the mobility increases. The success rate decreases and the performance of the scheme depends more on percentage of malicious nodes in the system. The number of clusters also do not change the fraction of success decisions.

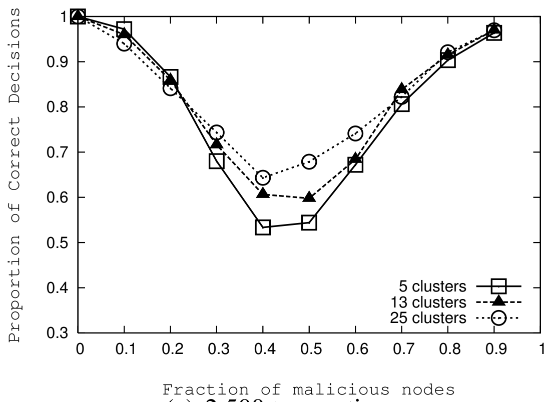
Basic reputation scheme **Token-based mechanism**



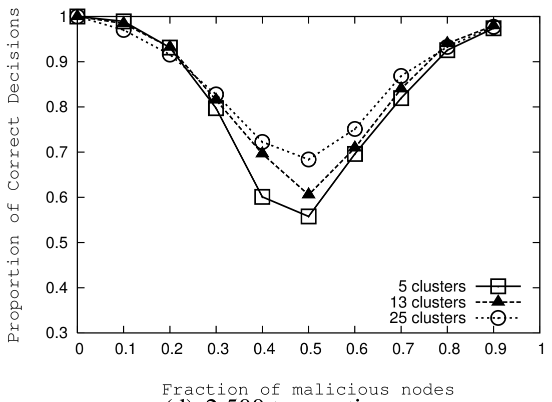
(a) 500 transactions



(b) 500 transactions



(c) 2,500 transactions



(d) 2,500 transactions

Figure 7.9: Proportion of correct decisions vs. proportion of nodes acting maliciously after $200,000$ iterations and medium speed.

Thus, we can conclude that small clusters enables the node to rely more on their direct experience, as they can exploit the amount of satisfaction in previous transactions to predict the behaviour of the nodes.

Moreover, the token helps nodes in taking decisions when they must evaluate the trustworthiness of newcomers.

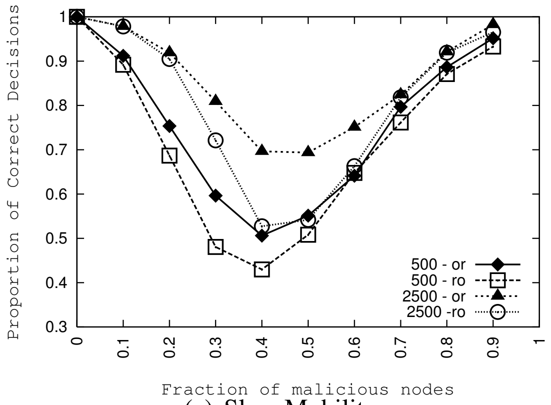
7.9.4 Proportion of malicious nodes

In Fig. 7.9 and 7.10 we plot the performance of our scheme with different percentage of malicious nodes in the network. As the proportion of malicious nodes in the network increases, we expect the proportion of correct decisions to decrease. In Fig. 7.10, the plots show the results for the different mobility settings for both the case of traditional reputation management systems and the token-based extension.

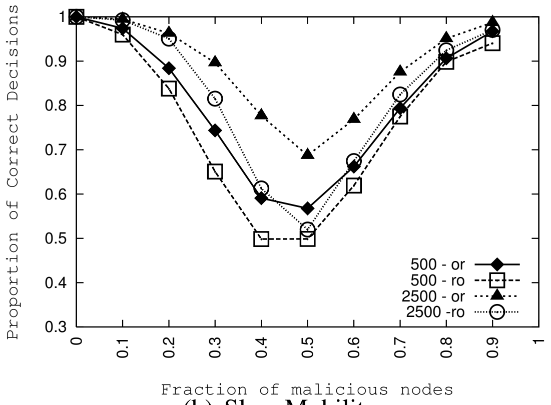
Fig. 7.10 illustrate the impact of malicious nodes on the number of clusters and the frequency of the movement. In all cases, the percentage of malicious nodes significantly impacts the proportion of correct decisions made. We are considering peers who act maliciously during the transaction and while providing reports on the transactions.

Thus, when the fraction of malicious nodes is greater than 50%, the success rate increases as malicious nodes reports the inverse of the amount of satisfaction they receive from an interaction. The feedbacks are aggregated at the designated agents that, if malicious, report the inverse of the estimated reputation value. Hence, if O (R) is the actual opinion (reputation) value, the value that is sent is $(1 - O)$ $((1 - R))$. This results in the subverting the feedbacks of the nodes. Correct decisions can be taken even if the malicious nodes dominate the system. The same behavior has been analyzed in [50] and in Section 6.5.3 for the ROCQ reputation scheme.

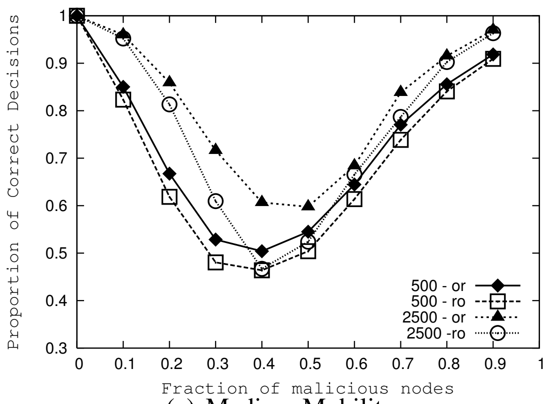
Basic reputation scheme **Token-based mechanism**



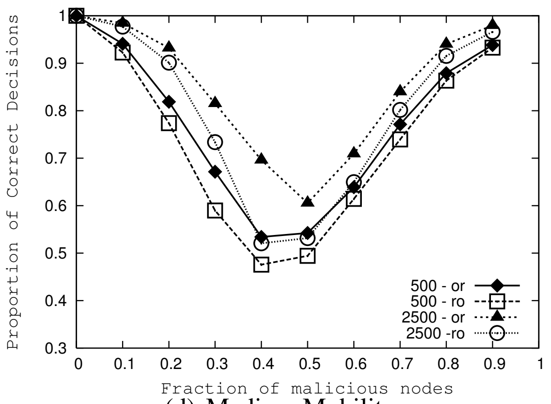
(a) Slow Mobility



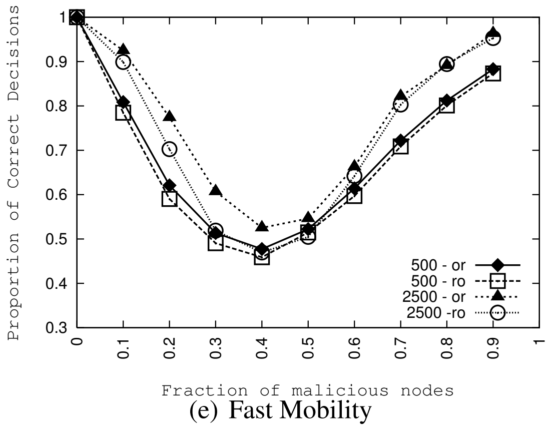
(b) Slow Mobility



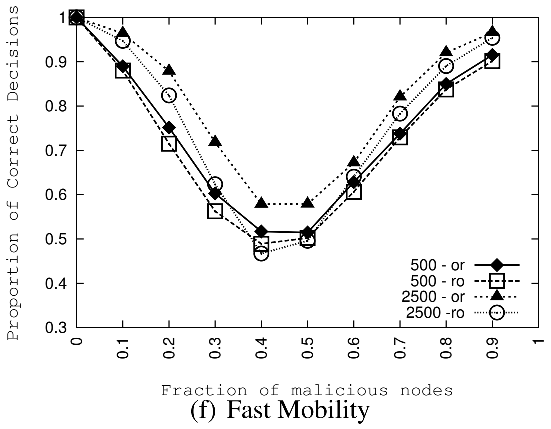
(c) Medium Mobility



(d) Medium Mobility



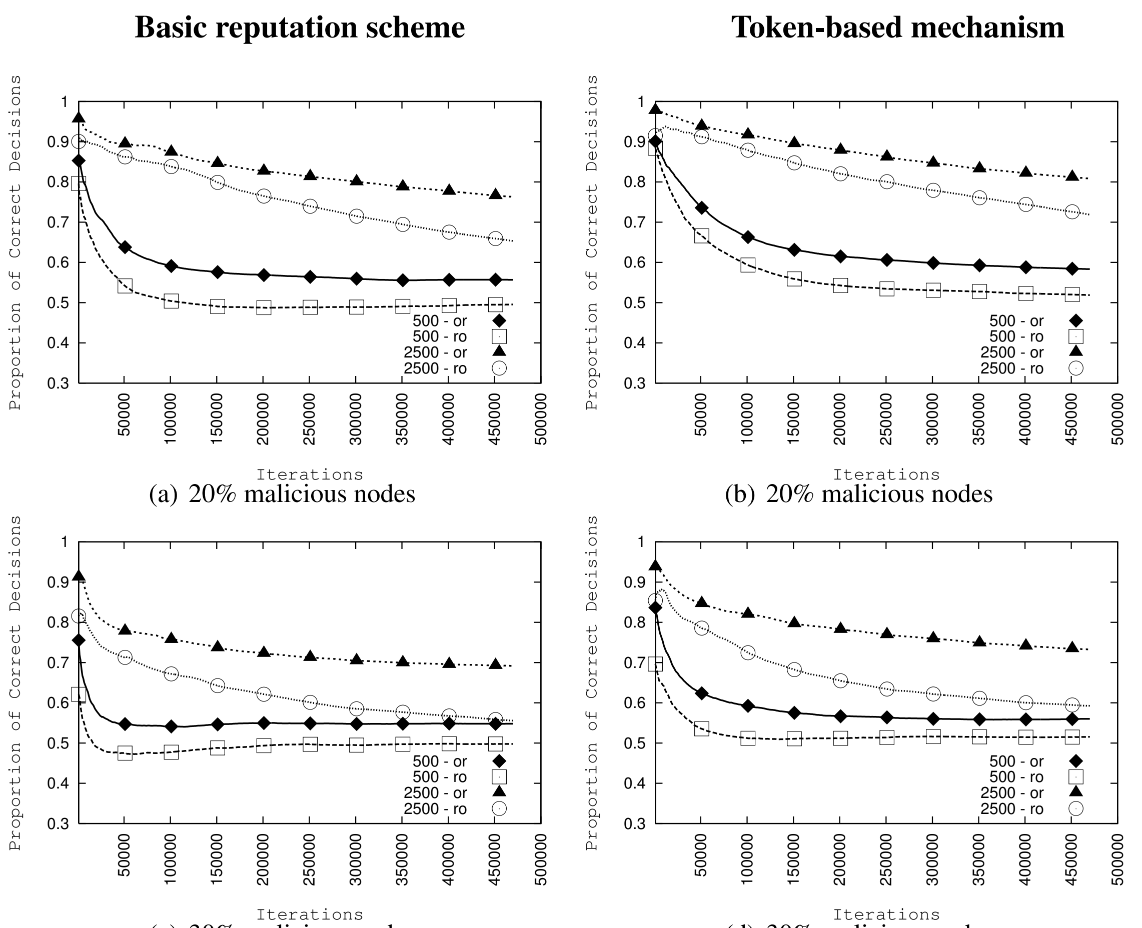
(e) Fast Mobility



(f) Fast Mobility

Figure 7.10: Proportion of correct decisions vs. proportion of nodes acting maliciously after $200,000$ iterations with 13 clusters.

7.9.5 Inconsistent behaviour



(a) 20% malicious nodes (b) 20% malicious nodes

(c) 30% malicious nodes (d) 30% malicious nodes

Figure 7.11: Proportion of correct decisions with slow mobility and 13 clusters when nodes have inconsistent behaviour during their lifetime.

When peers act maliciously in a consistent fashion while moving across clusters, it should be possible to identify them relatively quickly. However, if peers choose to act maliciously only in some clusters and for some transactions, it may be more difficult to identify them. This type of attack, known as *milking* and defined in Section 2.3.2, causes the credibility of the clusters to be lowered as they do not provide accurate information if nodes change behaviour.

We simulate that a fraction of nodes, equal to the percentage of malicious

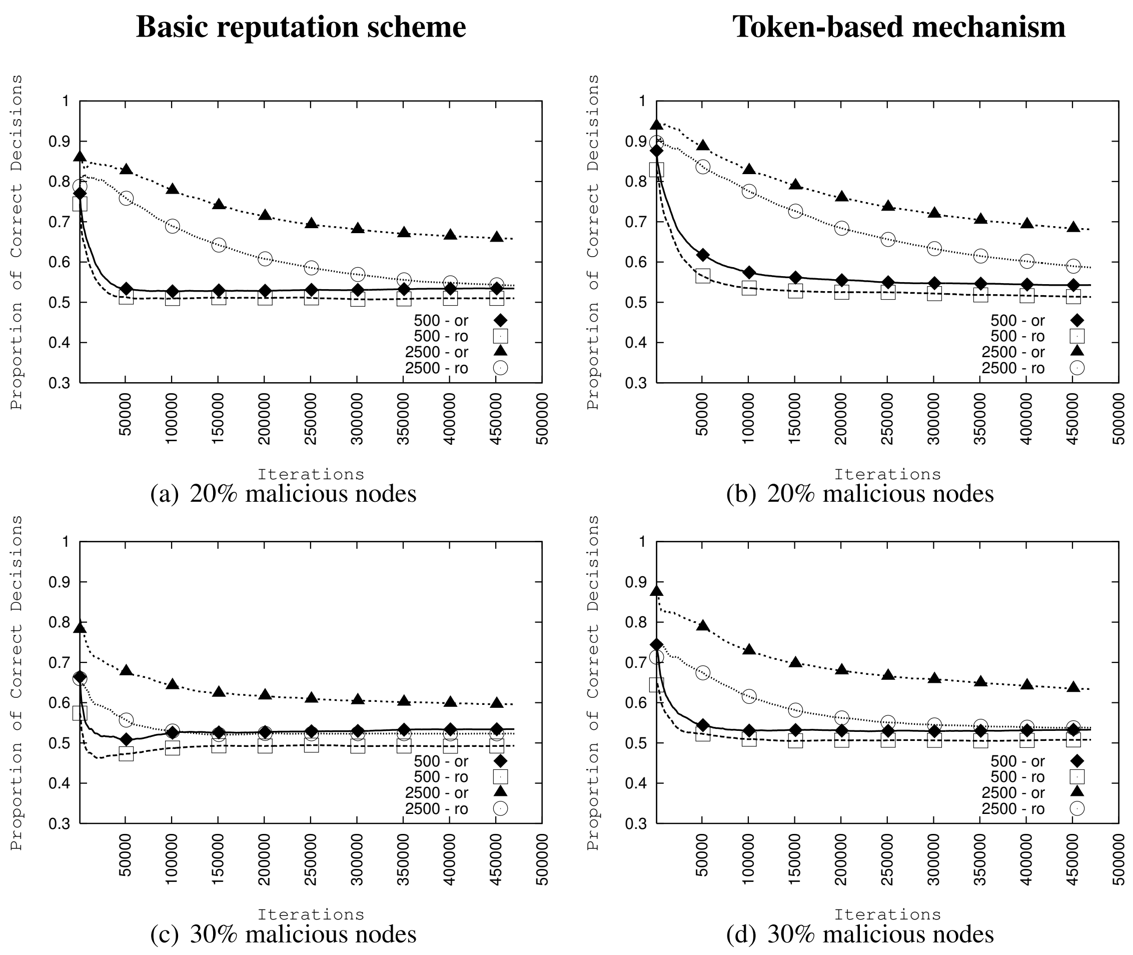


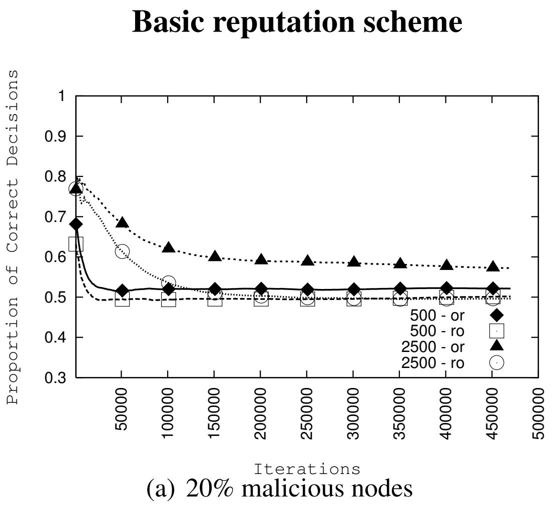
Figure 7.12: Proportion of correct decisions with medium mobility and 13 clusters when nodes have inconsistent behaviour during their lifetime.

nodes, changes behaviour with a rate that is half of the frequency of movements, which is given by the number of transactions between two subsequent movements. In Fig. 7.11 we plot the success rate when nodes move slowly in the system. A small percentage of nodes, that are initially malicious, decreases the performance of the system, in particular when the nodes uses reputation to decide their behaviour in a transaction, as shown in plots (a) and (b) in Fig. 7.11. The milking attack is more effective when the number of transactions between two subsequent movement is high, i.e., $2,500$. In this case, nodes form a consistent opinion of the members' trustworthiness and they put more confidence in their reports, sent to designated agents. Thus, when a node changes behaviour, its attack has higher impact on the performance of the system as the node might have acquired *privileges* for being cooperative before.

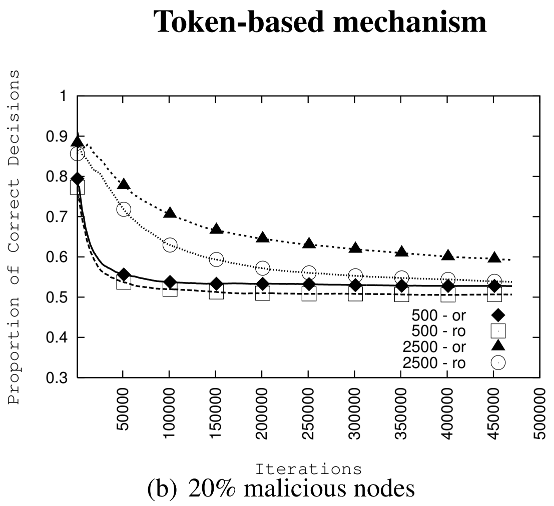
When the percentage of malicious nodes increases and when there are few transactions before each movement, the gain of the token-mechanism compared to the use of the traditional ROCQ reputation management is limited, plots (c) and (d) in Fig. 7.11. This is due to the low confidence value that the node has on the clusters that issue reports. If nodes change behaviour, clusters that have scored these nodes as cooperative (or malicious in the opposite case) have low credibility for their following reports. Thus, for following entering agents, designated agents put low confidence in the initial reputation value estimated from the reports in the token. This explains why the token-based approach gives little improvement, as new transactions count more in the evaluation of the trust value.

If we increase the speed of the nodes, the reputation management system cannot score nodes effectively. Fig. 7.12 and Fig. 7.13 plot the proportion of correct decisions for medium and fast mobility respectively. The performance of the token-based mechanisms are similar to the traditional reputation management system, when the percentage of malicious nodes increases, plots (c) and (d) in Fig. 7.12 and Fig. 7.13. However, with 20% of initial malicious nodes, the token-based mechanism performs better, for $2,500$ transactions before changing the clusters' membership, until the cluster confidence is such that designated agents cannot aggregate the reports correctly. This is shown by the decline of curves in plots (b) of Fig. 7.12 and Fig. 7.13.

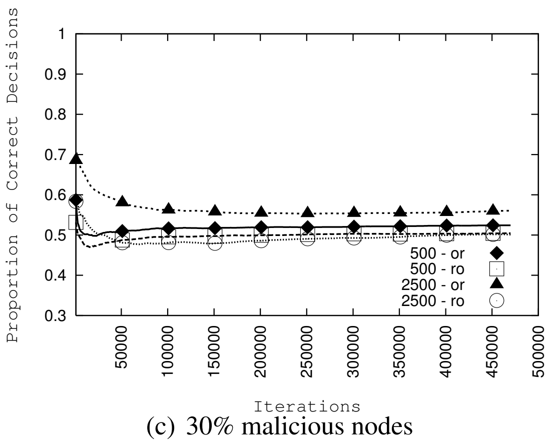
In Fig. 7.14 we plot the success rate of the reputation management scheme without and with the token-based extension for different number of clusters in the area. When the number of transaction is low, plots (a) and (b), the number of clusters do not have impact on the proportion of correct decisions. This behaviour can be attributed to the small number of transactions on which the



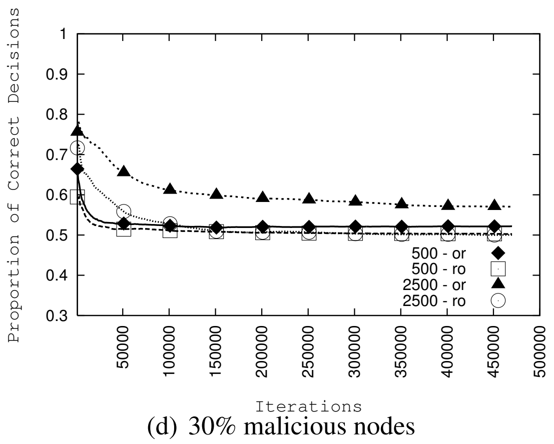
(a) 20% malicious nodes



(b) 20% malicious nodes



(c) 30% malicious nodes



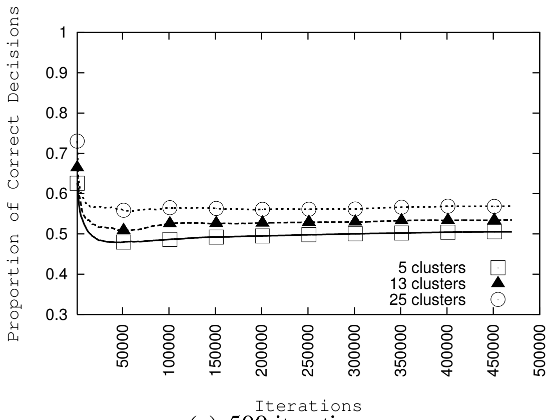
(d) 30% malicious nodes

Figure 7.13: Proportion of correct decisions with fast mobility and 13 clusters when nodes have inconsistent behaviour during their lifetime.

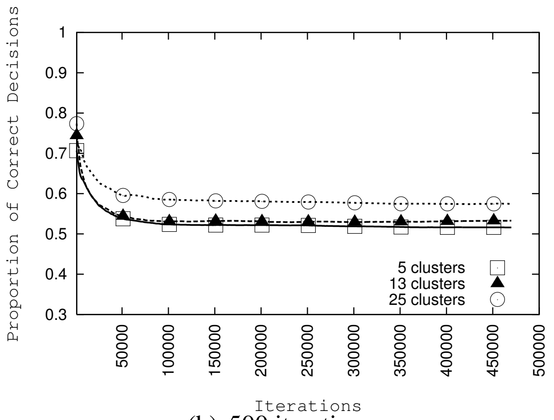
reputation value is formed. Moreover, the same pair of nodes hardly interact when there are few clusters as they are densely populated. This implies that nodes have inconsistent behaviour in these transaction.

In Fig. 7.15 and Fig. 7.16 we plot the performance of the reputation management schemes with different percentage of malicious nodes in the network. As observed earlier for a consisted behaviour of the nodes, the proportion of correct decisions decreases as the proportion of malicious nodes in the network increases. It is worth to notice that when the fraction of malicious nodes is greater than 50%, the percentage of correct decisions remains low in contrast

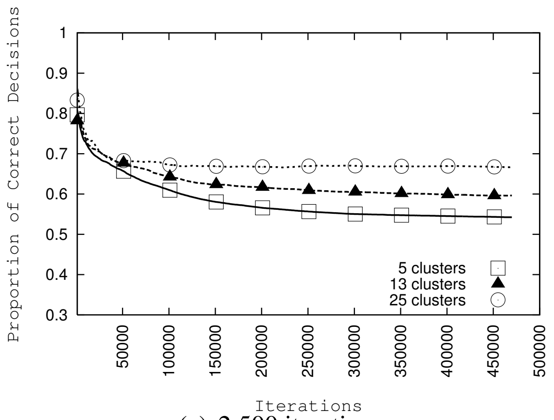
Basic reputation scheme **Token-based mechanism**



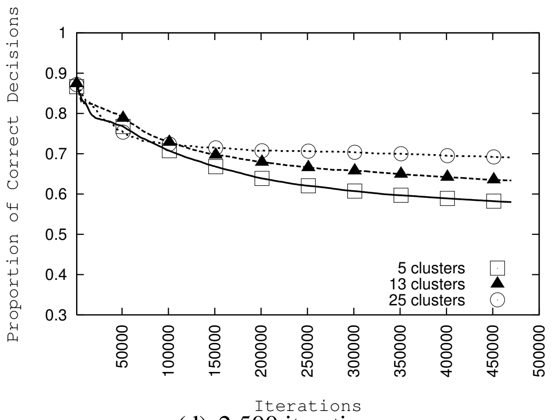
(a) 500 iterations



(b) 500 iterations



(c) 2,500 iterations

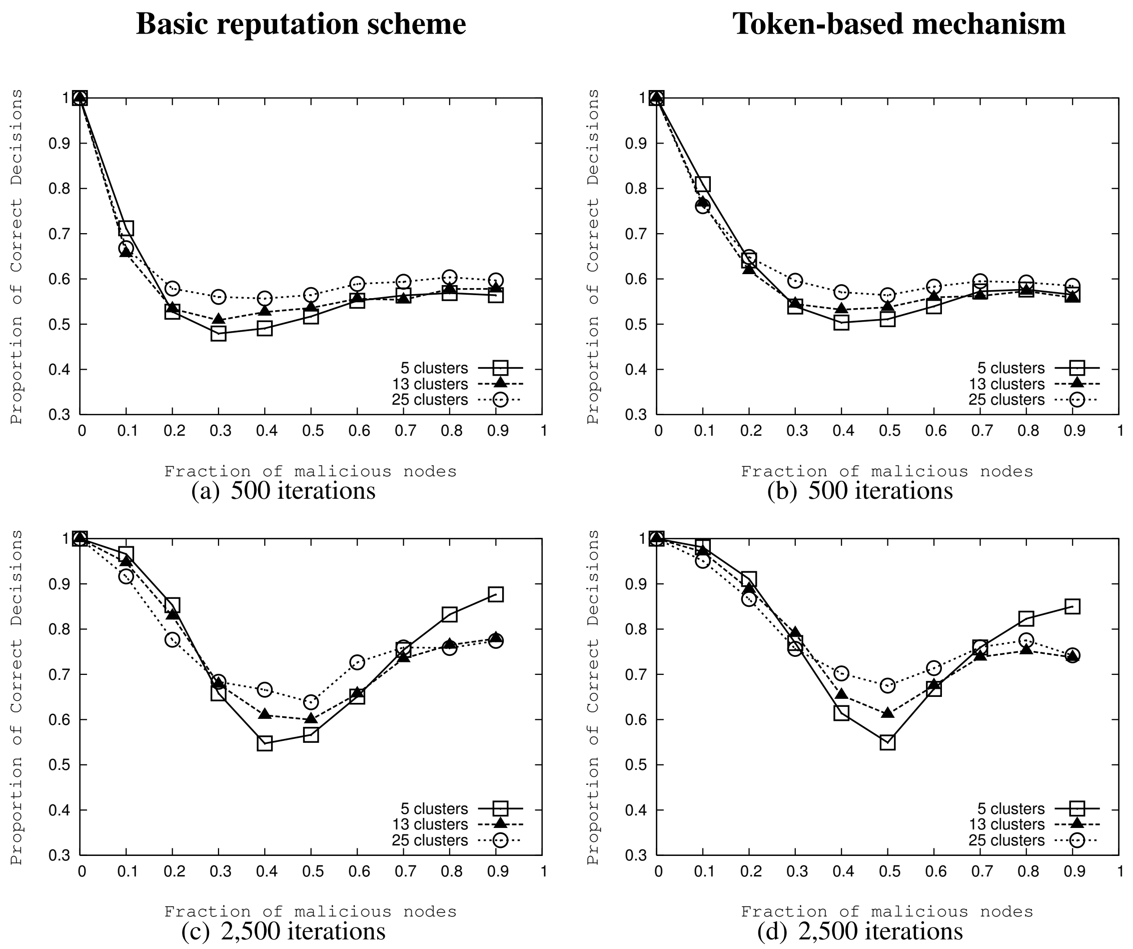


(d) 2,500 iterations

Figure 7.14: Proportion of correct decisions for medium mobility of the nodes and 30% of malicious nodes when nodes' behaviour is inconsistent.

with what we have observed in Fig. 7.10. This is valid when either nodes use reputation for the evaluation of the trustworthiness of a node or, in general, the number of transactions is low, i.e., 500. The constant success rate in Fig 7.16 can be attributed to the fact that the reputation values cannot predict the behaviour of the nodes correctly when nodes milk their behaviour. This is independent on the speed of the nodes.

Fig. 7.16 also shows that the token-based mechanism performs better than traditional reputation schemes when the fraction of malicious nodes is below 50%. With a low fraction of malicious nodes, the token helps to detect malicious

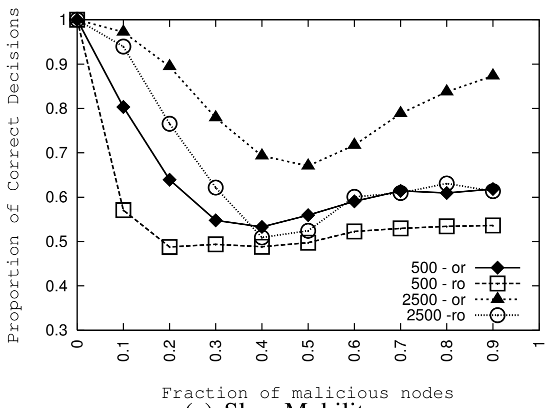


(a) 500 iterations (b) 500 iterations (c) 2,500 iterations (d) 2,500 iterations

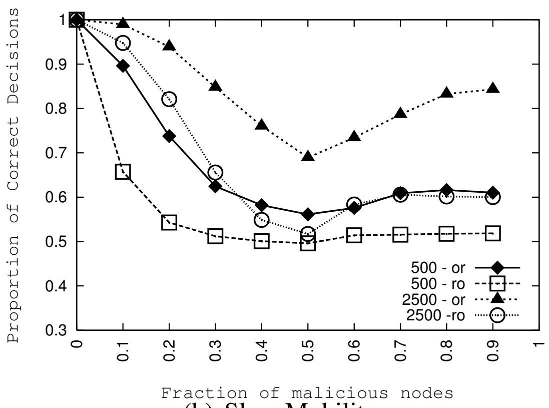
Figure 7.15: Proportion of correct decisions vs. proportion of nodes acting maliciously after $200,000$ iterations and medium speed.

nodes immediately. But, when malicious nodes are dominant in the system, the reliability of the token is low as the behaviour of the nodes is unpredictable and, since the credibility of the clusters becomes low, the token does not give an accurate view of nodes' maliciousness. If the number of clusters varies, Fig. 7.15, the performance of the reputation schemes do not change significantly.

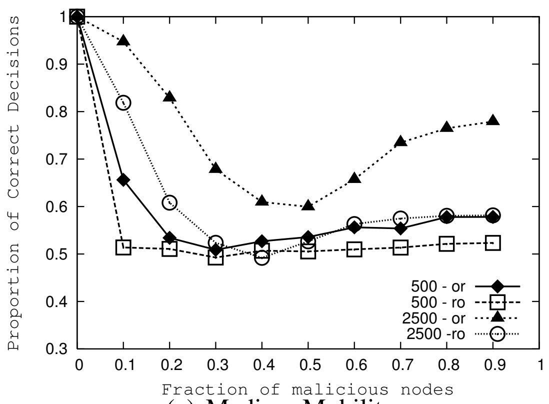
Basic reputation scheme **Token-based mechanism**



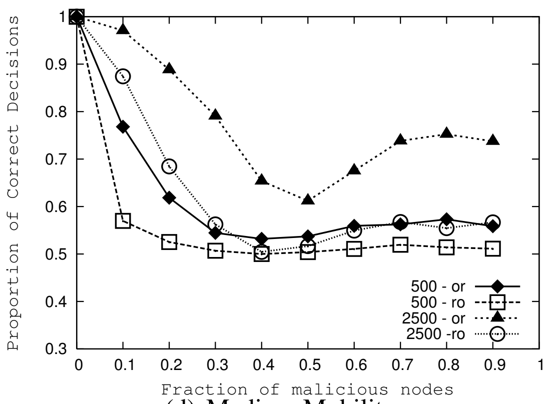
(a) Slow Mobility



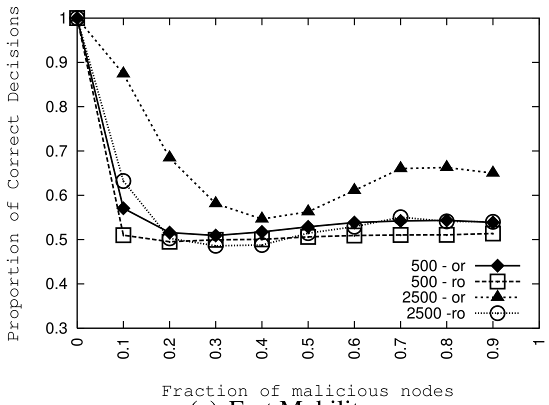
(b) Slow Mobility



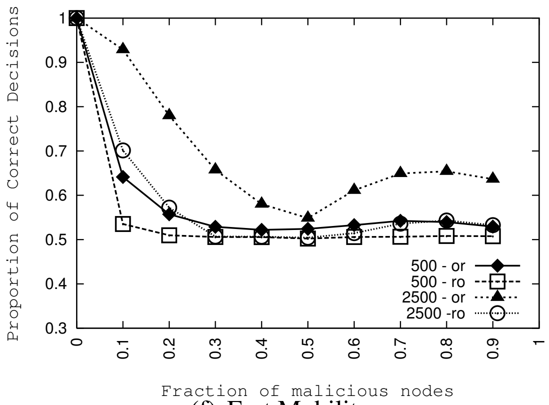
(c) Medium Mobility



(d) Medium Mobility



(e) Fast Mobility



(f) Fast Mobility

Figure 7.16: Proportion of correct decisions vs. proportion of nodes acting maliciously $200,000$ iterations with 13 clusters.

7.9.6 The case of *known* peers in communities

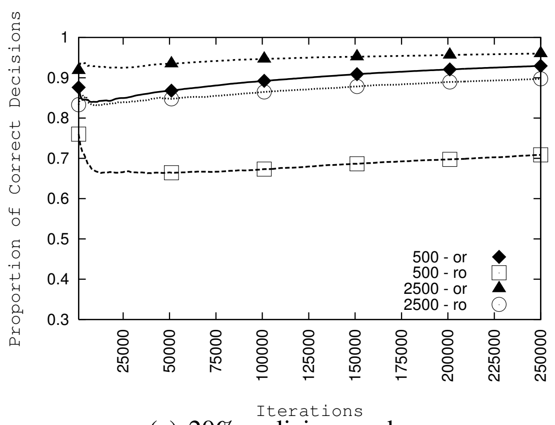
In this section, we evaluate the performance of the token-based extension when nodes maintain the history of their past transactions. In this case, the nodes do not delete the opinions formed on the amount of satisfaction they receive from other peers, as this information can be used if they meet the same node in other clusters.

The scenario that we considered in previous sections differs from the one studied herein as the former is based on the assumption that nodes might sporadically meet in the future. In fact, by cleaning the complete history of the nodes, every time a node joins a cluster, it is considered as a *stranger*. With these settings, the results, that we have obtained earlier, can be generalized to justify the effectiveness of the token-based scheme in a high dynamic environment where thousands of nodes can join or leave communities.

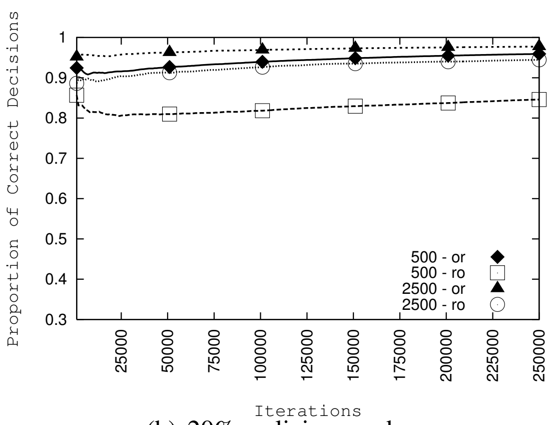
In many situations, communities are formed by nodes that can reveal similar mobility patterns or interests that are driven by daily habits. Thus, they can meet each other and interact while they join different communities. To analyze this scenario, we simulate a system with designated agents who delete the reputation values of the peers, that they are responsible for. As the membership of the community can change and the reputation values of the nodes are stored anonymously, designated agents are not required to maintain this information if a node leaves the community.

In Fig. 7.17, we plot the success rate for the traditional reputation management scheme (plots (a) and (c)) and the token based extension (plots (b) and (d)). As evident from the curves, in all cases, the performance of the two reputation schemes are comparable when opinion is used as evaluation metric for the decision, i.e. *or* in the plots. This is caused by the fact that reputation values are used sporadically to estimate the trustworthiness of a node, as direct experience is always available. In fact, when reputation is used to evaluate nodes'

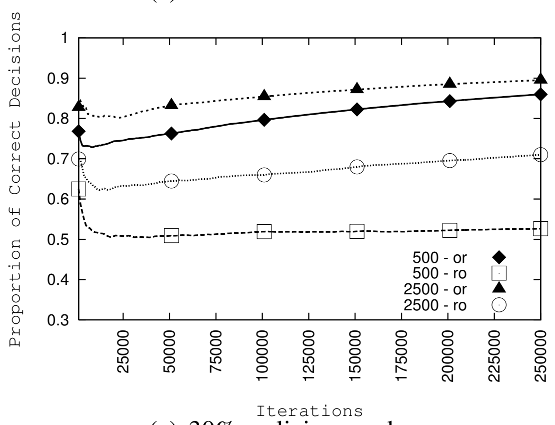
Basic reputation scheme **Token-based mechanism**



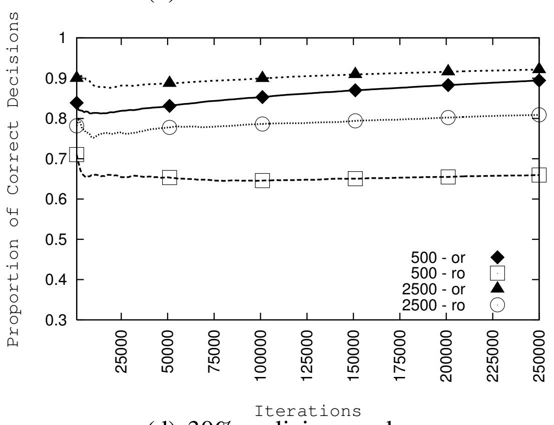
(a) 20% malicious nodes



(b) 20% malicious nodes



(c) 30% malicious nodes



(d) 30% malicious nodes

Figure 7.17: Proportion of correct decisions with medium mobility and 13 clusters.

trustworthiness, the token-based mechanism results in up to 15% more correct decisions than when no mobility extension is used. The same increase is present in Fig 7.5. Similar conclusions can be derived for different speed of the nodes independently.

Fig. 7.18 shows the impact of the number of clusters on the performance when opinion is used first as metric to judge nodes' behaviour. Even if there are few transactions between two subsequent movements, more clusters allow nodes to interact frequently and form more accurate opinions (recall that the nodes can only interact within the cluster). Thus, the token-based extension

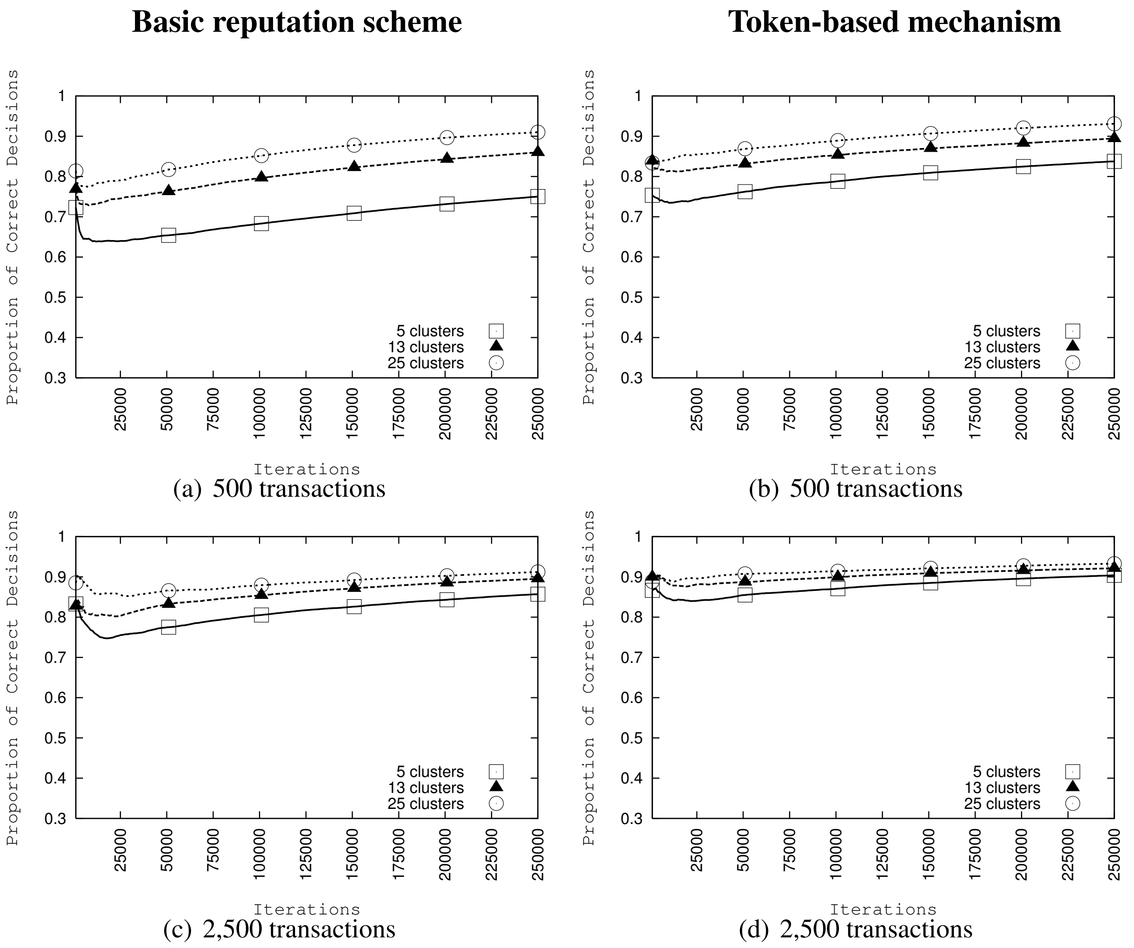
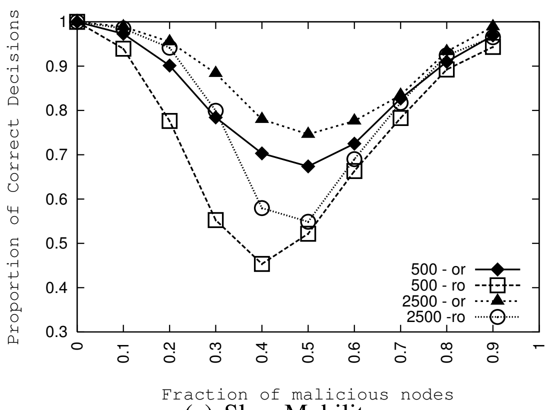


Figure 7.18: Proportion of correct decisions when nodes maintan opinion with medium mobility of the nodes and 30% of malicious nodes

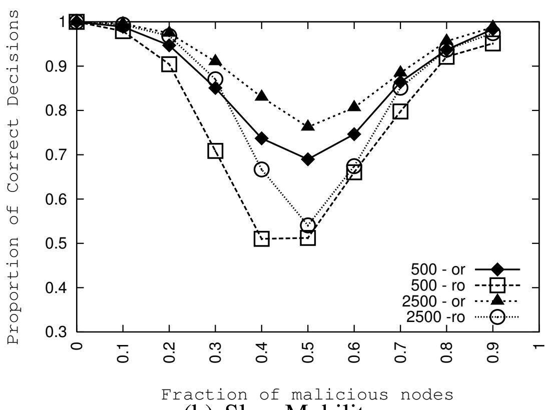
does not increase the success rate when there are 13 or 25 clusters, as shown in plots (a) and (b) of Fig. 7.17. But, if there are 5 clusters, nodes meet less frequently and the token-based approach can compensate the lack of opinions.

When the number of transactions increases, we can expect the performances of the schemes with a different number of clusters to be comparable, as depicted in plots (c) and (d) of Fig. 7.18, because nodes have a sufficient number of samples to calculate form opinions on the nodes.

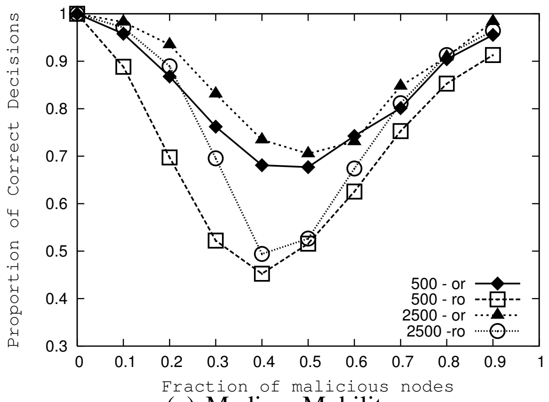
Basic reputation scheme **Token-based mechanism**



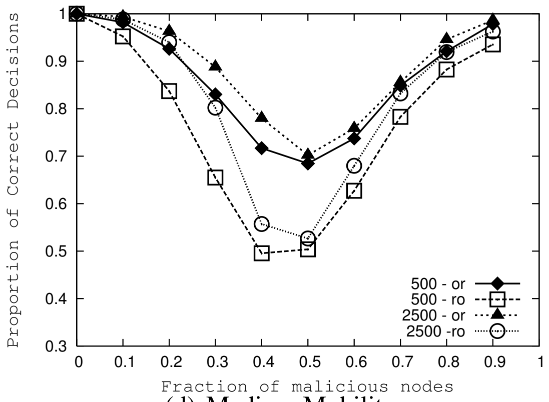
(a) Slow Mobility



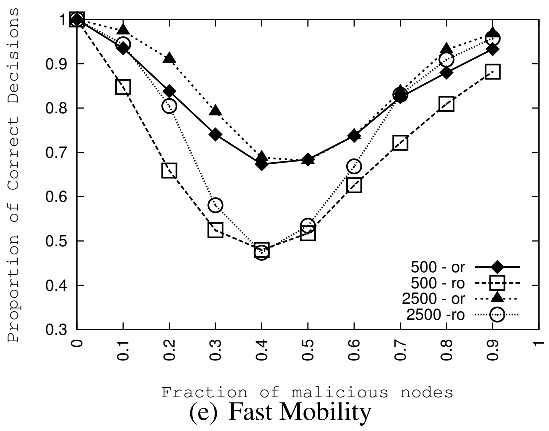
(b) Slow Mobility



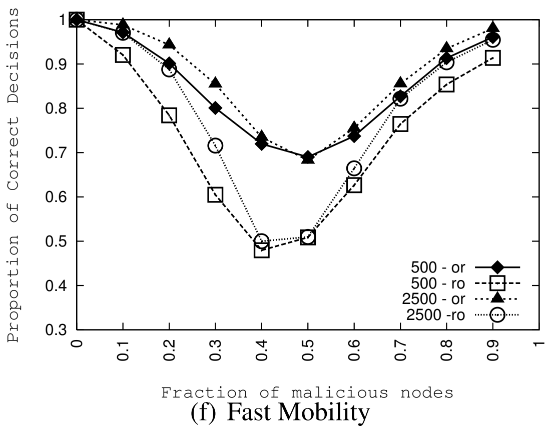
(c) Medium Mobility



(d) Medium Mobility



(e) Fast Mobility



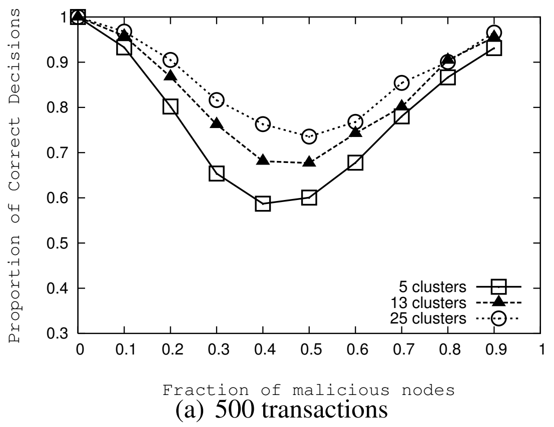
(f) Fast Mobility

Figure 7.19: Proportion of correct decisions vs. proportion of nodes acting maliciously after $200,000$ iterations and 13 clusters.

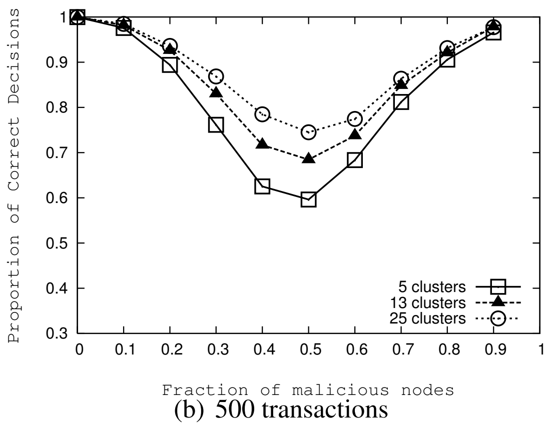
Proportion of malicious nodes

In Fig 7.19, we plot the success rate as function of the percentage of malicious nodes for different mobility scenarios. As noticed earlier, the token-based solution and the traditional reputation management system have similar performances when nodes base their decisions only on opinion.

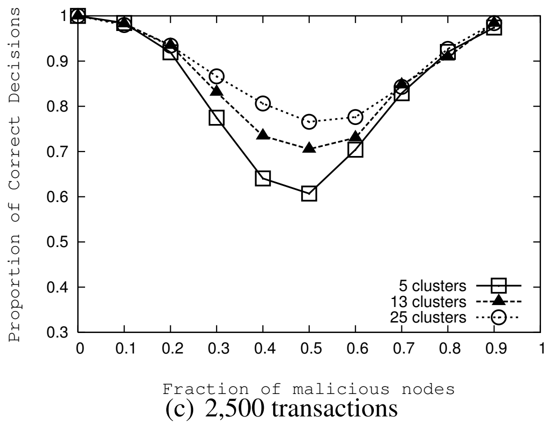
Basic reputation scheme **Token-based mechanism**



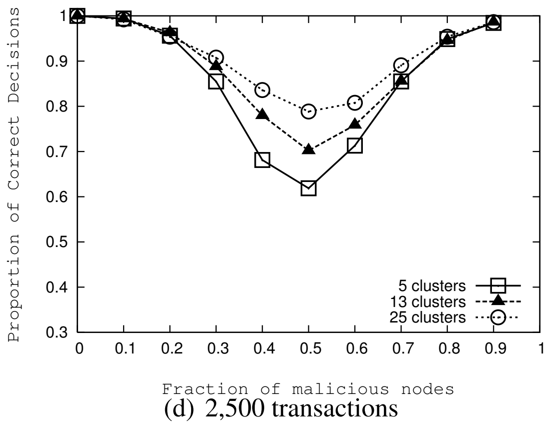
(a) 500 transactions



(b) 500 transactions



(c) 2,500 transactions



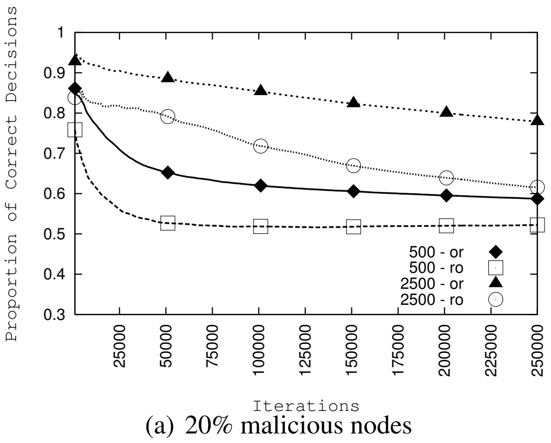
(d) 2,500 transactions

Figure 7.20: Proportion of correct decisions vs. proportion of nodes acting maliciously when the nodes move with medium speed after $200,000$ iterations.

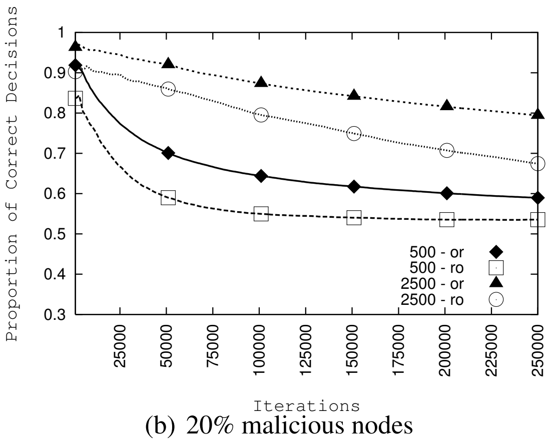
However, when reputation is used, the token helps the communities to predict with more accuracy the nodes' trustworthiness when the percentage of malicious nodes increases upto 50%. This is more evident for a system characterized by frequent changes of the topology due to the high speed of the nodes, plots (d) and (f) in Fig 7.19.

The plots in Fig. 7.20 show the results of the simulations with medium mobility of the nodes and opinion as metric to evaluate trust values. This case is the most unfavorable to evaluate the gain of the token-based extension as this extension to support mobility improves the accuracy of the reputation values and their timely estimation. The token has no effect on the definition of the opinion as it is based on the personal experience of the nodes only. For the case of 500 transactions and percentage of malicious nodes below 50%, the token-based extension increases the success rate for different numbers of clusters, see plots (a) and (b) in Fig. 7.20. In particular, it is evident in plot (b) that with 5 clusters there are 10% more correct decisions on average when the token is used.

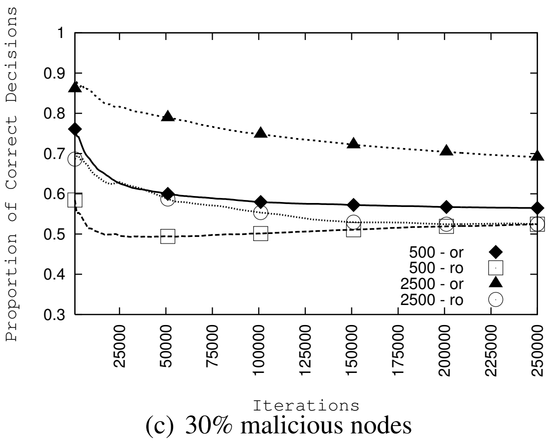
Basic reputation scheme **Token-based mechanism**



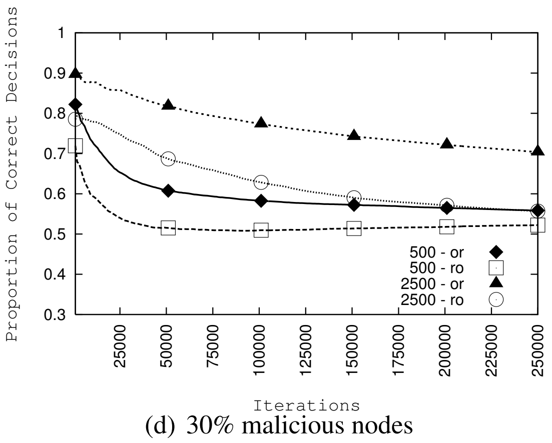
(a) 20% malicious nodes



(b) 20% malicious nodes



(c) 30% malicious nodes



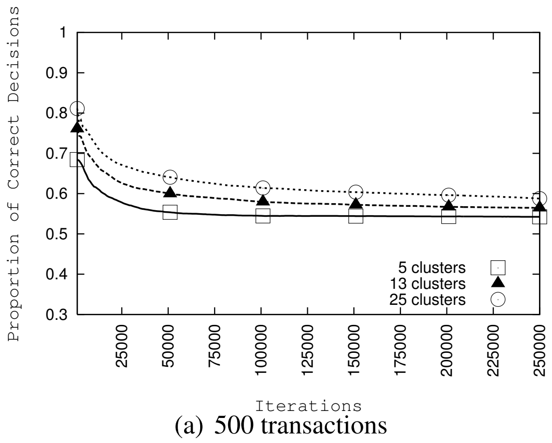
(d) 30% malicious nodes

Figure 7.21: Proportion of correct decisions with medium mobility of the nodes and 13 clusters.

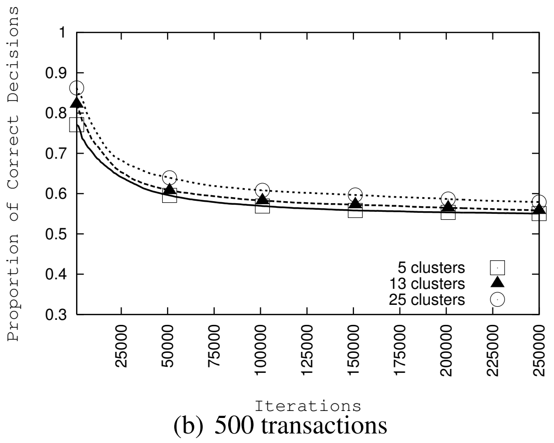
Inconsistent behaviour

We now examine the effect of the milking attack in the *known peers community* scenario. In Fig. 7.21 we plot the proportion of correct decisions with 20% (plots (a) and (b)) and 30% (plots (c) and (d)) of nodes that are initially malicious. The inconsistent behaviour of the nodes decreases the performance of the system, in particular, when the nodes use reputation to decide their behaviour in a transaction, see plots (a) and (b) in Fig. 7.21. If there are 500 transactions before each movement, the reputation management system cannot react effectively to the presence of traitors as the nodes cannot form consistent opinions.

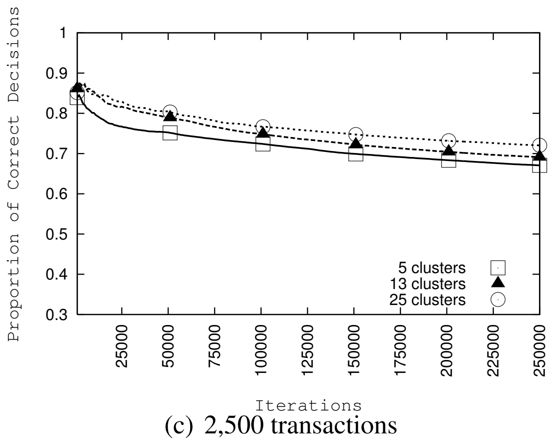
Basic reputation scheme **Token-based mechanism**



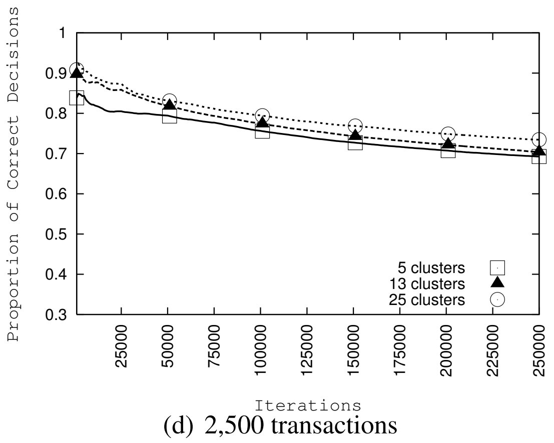
(a) 500 transactions



(b) 500 transactions



(c) 2,500 transactions



(d) 2,500 transactions

Figure 7.22: Proportion of correct decisions with medium mobility of the nodes and 30% of malicious nodes.

However, when nodes interact for $2,500$ iterations before the community is reorganized, the token-based extension is able to give better performance if reputation is used to take decision on the trustworhiness of nodes, plots (a) and (b) in Fig. 7.21. If we increase the percentage of malicious nodes up to 30%, the gain of the token-mechanism compared to the use of traditional reputation management is marginal, plots (c) and (d) in Fig. 7.21. This is due to the low confidence the node has on the clusters that issue reports.

Fig. 7.22 show the success rate for different number of clusters. In (a) and (c), we plot the performance of the traditional reputation management scheme

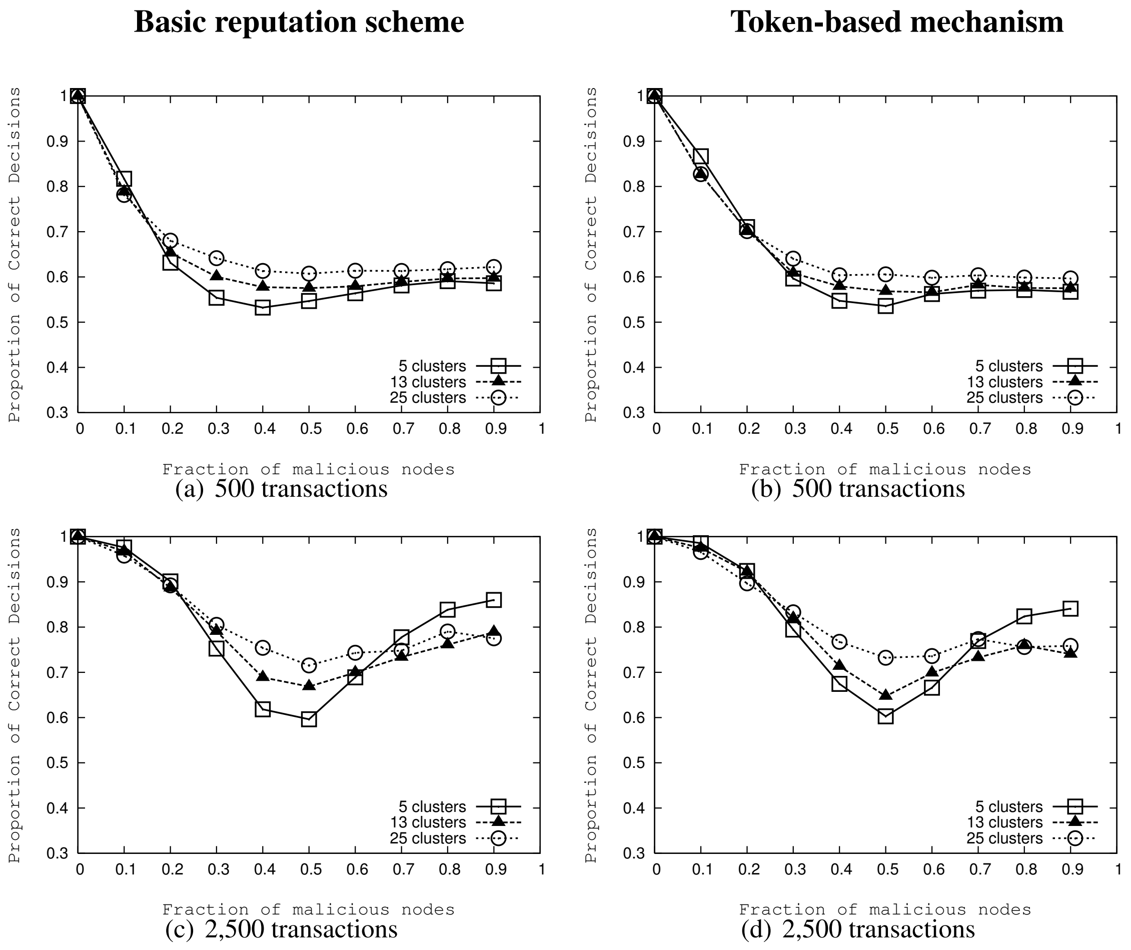
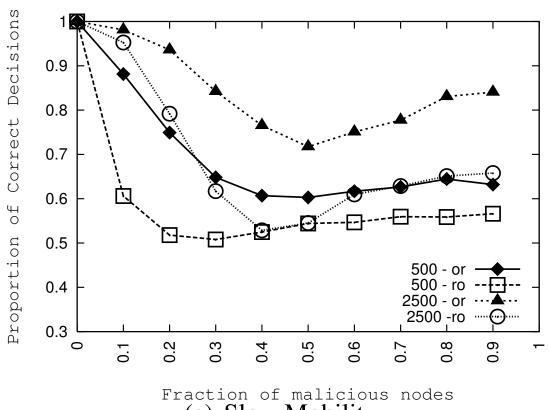


Figure 7.23: Proportion of correct decisions vs. proportion of nodes acting maliciously after $200,000$ iterations when the nodes move with medium speed.

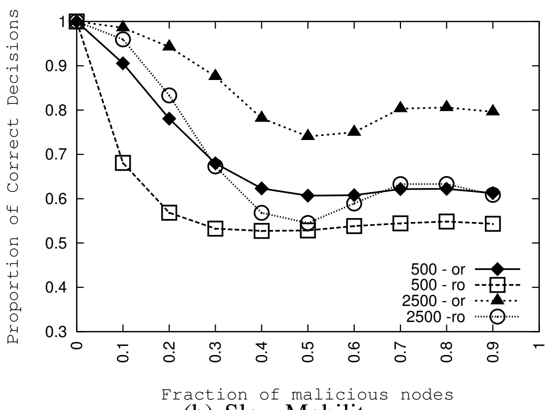
and in (b) and (d) the token-based extension. The number of clusters do not impact on the fraction of correct decisions for both schemes. This is because nodes have an alternate behaviour, thus, the number of direct transactions, as result of the number of clusters, does influence the confidence nodes that have in their opinions as the quality value is low due to the inconsistent maliciousness. This also justifies the decrease of the success rate in all cases.

In Fig. 7.23 and Fig. 7.24 we plot the results of the simulations with different percentage of malicious nodes. The curves show that same behaviour observed in Section 7.9.5. This is due to the inconsistent behaviour of the nodes.

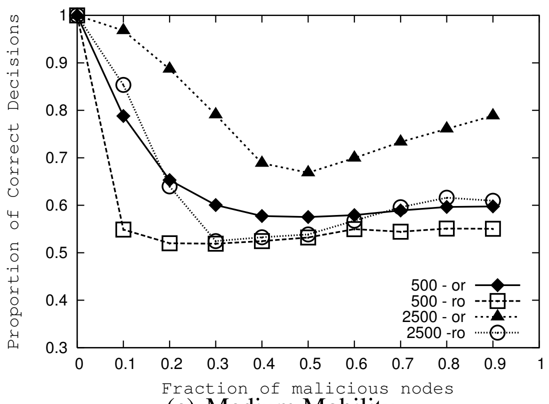
Basic reputation scheme **Token-based mechanism**



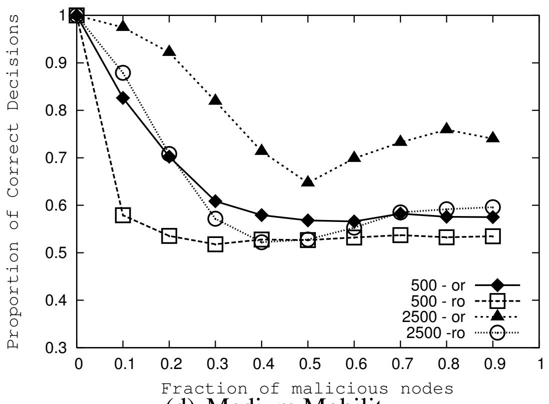
(a) Slow Mobility



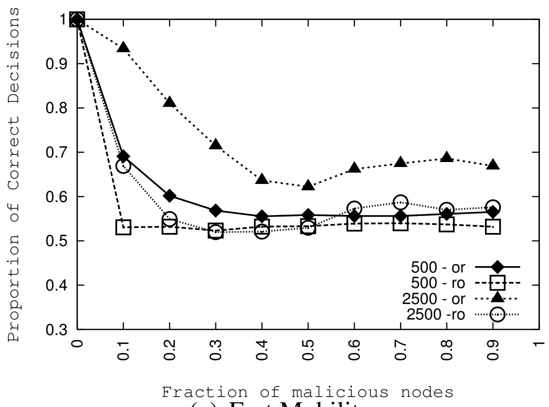
(b) Slow Mobility



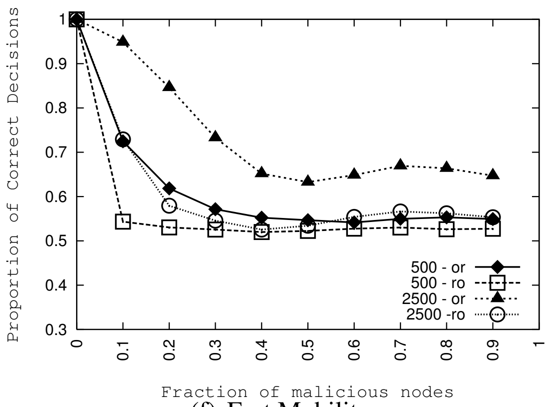
(c) Medium Mobility



(d) Medium Mobility



(e) Fast Mobility



(f) Fast Mobility

Figure 7.24: Proportion of correct decisions vs. proportion of nodes acting maliciously after $200,000$ iterations and 13 clusters in the system.

The fact that in the *known peers community* scenario, nodes have interacted in the past and have formed a view of the nodes' trustworthiness do not help them in taking an accurate decision. The feedbacks are inconsistent and nodes do not put a high confidence in their reports.

Thus, either the number of clusters or the speed of the nodes do not change the performance of the reputation management schemes, as the schemes are not resilient to milking attacks if a node does not behave consistently for a fraction of transactions. This can be observed in Fig. 7.24 for $2,500$ transactions when opinion is used to evaluate if nodes are trustworthy.

Chapter 8

Conclusion and future work

Autonomic communication systems have inherent self-organizing and self-scalability properties that make their applicability suitable in different contexts, ranging from content distribution networks to context-aware service application. These systems incorporate methods to monitor their dynamic behaviour and to react in automated ways in order to effectively reconfigure themselves to adapt to new environments. These concepts of self-management and self-adaptation are fully applicable in all decentralized peer to peer systems or mobile ad-hoc networks.

In autonomic systems, the intelligence and the monitoring capabilities required to guarantee the correct execution of the protocol are shared by the nodes forming these networks. However, relying on the nodes to make the system functioning properly introduces new threats which derive from the uncooperative behaviour of the nodes. In un-managed and distributed systems, reputation management schemes are effective in eliminating malfunctioning component and in fostering cooperation among selfish nodes.

In this thesis we justify the use of the reputation management schemes to self-preserve autonomic communication systems and we analyze the applicability of these schemes in specific contexts.

We begin with an overview of autonomic communication systems and with a discussion of the new security paradigms and threats in these system. The goal is to analyze the security requirements of autonomic systems to facilitate the evaluation of the solutions proposed in the literature. We focus on reputation management schemes as they can thwart malicious and uncooperative nodes' behaviour.

In Chapter 3 we define the architecture solutions and the basic components required for the design of reputation management systems. We discuss security issues and we critically assess the suitability of existing techniques in autonomic communication systems. In our discussion, we describe the metrics for an extensive evaluation of the performance of reputation schemes.

In Chapter 4, we evaluate the communication overhead of reputation systems and we discuss the benefit of using reputation when compared with the cost of extra-signalling. We find that the design of reputation schemes must take in consideration the underlying network topology and the application for which the scheme is deployed. The communication overhead can cause network congestion if the schemes to collect feedbacks and disseminate reputation overwhelm the normal functionality of the application. A hybrid approach is more suitable in a dynamic environment, as it must adapt to the context of communication.

In Chapter 5 we propose a game theoretic approach based on the Iterated Prisoner's Dilemma to study reputation management systems. We derive the *Nash equilibrium* for the reputation game and we show that reputation is sufficient to sustain cooperation if players are *patient*. We also give a quantitative estimation of the time a node must wait before being rewarded for its cooperative behaviour. We also assess the performance of the reputation scheme in the presence of nodes that always defect. Results show that reputation is needed to isolate faulty components. Unlike other approaches we define a framework to study how reputation is valuable for a node and we analyze the evolution of reputation in the system. The results of this work can be generalized to any application as we exploit the concept of reputation and use a simplified and general reputation management scheme.

Next, we concentrate on more realistic application of reputation by proposing an extension to the ROCQ reputation management scheme to study reputation in collaborative content distribution networks. We define the type of malicious nodes that are active in content distribution systems and we show the advantage of using reputation management schemes in a content distribution network through simulations. Our results show that reputation schemes improve the proportion of correct downloads by up to 16% and performs well both when the sender of the corrupt blocks can be identified and when he cannot be identified.

Finally, we propose a mechanism to extend the use of reputation in mobility scenarios. We present a token-based solution, that extends the functions of general reputation management systems, and allows nodes to carry information on their reputation defined in different contexts and communities. We discuss the security threats specific to a dynamic setting when nodes meet sporadically and form ad hoc communities to exchange content or context information. This token-based extension moves the burden of storing personal information to the nodes and enables correlation of reputation values earned in different communities. We demonstrate the effectiveness of token to eliminate the problem of the bootstrap of reputation in a new community and to timely detect malicious nodes. Nodes can exploit their reputation to immediately benefit services in a new community. Through simulations, we show that the token-based solution increases the success rate by 15% in the presence of 30% misbehaving nodes.

We end by presenting research areas that build upon the work described in this thesis.

Service reputation

The game theory model defined in Chapter 5 could be refined to apply the concept of reputation to service applications. Nodes should not value only the amount of satisfaction received by other nodes after an interaction, but also if their participation in the system is beneficial to them. The next step would be to define a game theoretical model that considers the amount of nodes' collaboration in the utility function to define the best strategy for the node in terms of shared resources. The reputation value is associated to the quantity of resources a node shares and the quality of the service provision. The theoretical results could be evaluated in real case scenarios for homogeneous and heterogeneous levels of cooperation.

Reputation in BitTorrent

The work presented in Chapter 6 and Chapter 7 can be combined to deploy a reputation management system suitable for BitTorrent. BitTorrent is a popular file sharing system that distributes files in blocks by using the concept of torrent. The implementation of BitTorrent considers the contribution level of nodes that participate in each torrent separately. This approach fasten the distribution of a single file but it does not help to transfer the reputation from one context to another.

We could apply a modified version of the token based mechanism to correlate torrents. By changing the algorithms that BitTorrent use to distribute blocks, we can assign higher priority to trusted peers. These peers should deploy a faster distribution of the content to other nodes. Moreover, reputation management schemes can thwart malicious nodes that distribute polluted content.

Reputation for vehicular communities

The high mobility of cars does not facilitate the definition and deployment of a suitable reputation management scheme. Vehicular networks are characterized by the ad hoc formation of communication links and in general small available bandwidth to transfer data. In this settings, the token-based mechanisms, presented in Section 7, could be applied in this highly dynamic context if the communities are organized in a way that they reflect the current networking system.

We cannot rely anymore on permanent designated agents for a node as the topology of the network and the communication links change continuously. Gossip-based approaches could be used to disseminate reputation in vehicular communication systems and the reputation could be stored by using the token-based mechanism.

Bibliography

[1] Athens wireless metropolitan network. http://www.awmn.net.

[2] Brooklyn museum wireless.
http://www.brooklynmuseum.org/visit/wireless.php.

[3] Canu mobility simulation environment (canumobisim).
http://canu.informatik.uni-stuttgart.de/mobisim/.

[4] Roma wireless. http://www.romawireless.com.

[5] Seattle wireless. http://www.seattlewireless.net.

[6] K. Aberer. P-grid: A self-organizing access structure for p2p information systems. In *CooplS '01: Proceedings of the 9th International Conference on Cooperative Information Systems*, pages 179–194, London, UK, 2001. Springer-Verlag.

[7] K. Aberer, P. Cudré-Mauroux, A. Datta, Z. Despotovic, M. Hauswirth, M. Punceva, and R. Schmidt. P-grid: a self-organizing structured p2p system. *SIGMOD Record*, 32(3):29–33, 2003.

[8] K. Aberer and Z. Despotovic. Managing trust in a peer-2-peer information system. In *Proceedings on the Tenth International C onference on Information and Knowledge Management (CIKM-01)*, pages 310–317, New York, November 5-10 2001. ACM Press.

[9] R. Ahlswede, N. Cai, S. Li, and R. W. Yeung. Network Information Flow. *IEEE Transactions on Information Theory*, 46(4):1204–1216, July 2000.

[10] A. S. Aiyer, L. Alvisi, A. Clement, M. Dahlin, J.-P. Martin, and C. Porth. Bar fault tolerance for cooperative services. *SIGOPS Operating Systems Review*, 39(5):45–58, 2005.

[11] E. Altman, T. Boulogne, R. El-Azouzi, T. Jimenez, and L. Wynter. A survey on networking games in telecommunications. *Computers and Operations Research*, 33(2):286–311, 2006.

[12] K. G. Anagnostakis and M. B. Greenwald. Exchange-Based Incentive Mechanisms for Peer-to-Peer File Sharing. In *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*, pages 524–533, Tokyo, Japan, March 23-26 2004. IEEE Computer Society.

[13] M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: formal definition, simplified requirements and a construction based on trapdoor permutations. In E. Biham, editor, *Advances in cryptology - EUROCRYPT 2003, proceedings of the internarional conference on the theory and application of cryptographic techniques*, volume 2656 of *Lecture Notes in Computer Science*, pages 614–629, Warsaw, Poland, May 2003. Springer-Verlag.

[14] M. Bellare, H. Shi, and C. Zhang. Foundations of group signatures: The case of dynamic groups. In *Topics in Cryptology CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science (LNCS)*, pages 136–153, 2005.

[15] E. W. Biersack, P. Rodriguez, and P. Felber. Performance Analysis of Peer-to-Peer Networks for File Distribution. In *The Fifth International Workshop on Quality of Future Internet Services (QofIS'04)*, Barcelona, September 29 - October 1 2004.

[16] D. Boneh and M. Franklin. Identity based encryption from the weil pairing. *SIAM Journal of Computing*, 32(3):586–615, 2003.

[17] D. Boneh and H. Shacham. Group signatures with verifier-local revocation. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, pages 168–177, Washington DC, USA, October 25-29 2004. ACM Press.

[18] E. Bresson, J. Stern, and M. Szydlo. Threshold ring signatures and applications to ad-hoc groups. In *CRYPTO '02: Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology*, pages 465–480, Santa Barbara, CA, USA, August 18-22 2002. Springer-Verlag.

[19] S. Buchegger and J.-Y. L. Boudec. Performance analysis of the confidant protocol. In *MobiHoc '02: Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 226–236, Lausanne, Switzerland, 2002.

[20] S. Buchegger and J.-Y. L. Boudec. A robust reputation system for P2P and mobile ad-hoc networks. In *Second Workshop on the Economics of Peer-to-Peer Systems*, Cambridge, MA, USA, 2004.

[21] C. Buragohain, D. Agrawal, and S. Suri. A game theoretic framework for incentives in p2p systems. In *Proceedings of the 3rd International Conference on Peer-to-Peer Computing (P2P '03)*, page 48, Linköping, Sweden, September 2003. IEEE Computer Society.

[22] L. Buttyan and J.-P. Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. *Mobile Networks and Applications*, 8(5):579–592, 2003.

[23] J. W. Byers, J. Considine, M. Mitzenmacher, and S. Rost. Informed Content Delivery Across Adaptive Overlay Networks. *IEEE/ACM Transactions on Networking*, 12(5):767–780, October 2004.

[24] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege. A Digital Fountain Approach to Reliable Distribution of Bulk Data. In *SIGCOMM*, pages 56–67, 1998.

[25] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. *Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2(5):483–502, 2002.

[26] R. G. Cascella. The “Value” of Reputation in Peer-to-Peer Networks. In *Fifth IEEE Consumer Communications and Networking Conference (CCNC 2008)*, Las Vegas, Nevada, USA, January 10-12 2008.

[27] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach. Secure routing for structured peer-to-peer overlay networks. In *Fifth Symposium on Operating Systems Design and Implementation (OSDI ’02)*, Boston, Massachusetts, December 2002.

[28] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh. SplitStream: High-bandwidth content distribution in cooperative environments. In *The 2nd International Workshop on Peer-to-Peer Systems (IPTPS ’03)*, Berkeley, CA, 2003.

[29] D. Charles, K. Jain, and K. Lauter. Signatures for network coding. In *40th Annual Conference on Information Sciences and Systems*, pages 857–863, Princeton, NJ, USA, March 2006.

[30] D. Chaum and E. van Heyst. Group signatures. In *Advances in Cryptology EUROCRYPT '91*, volume 547 of *Lecture Notes in Computer Science (LNCS)*, pages 257–265, Brighton, UK, April 1991.

[31] S. S. Chow, L. C. Hui, and S. Yiu. Identity based threshold ring signature. In *Information Security and Cryptology ICISC 2004*, volume 3506 of *Lecture Notes in Computer Science*, pages 218–232, 2005.

[32] B. Cohen. Incentives Build Robustness in BitTorrent. In *1st Workshop on Economics of Peer-to-Peer Systems*, Berkeley, CA, USA, June 5-6 2003.

[33] E. Damiani, S. De Capitani Di Vimercati, S. Paraboschi, and P. Samarati. Managing and sharing servants' reputations in P2P systems. *IEEE Transactions on Data and Knowledge Engineering*, 15(4):840–854, July-Aug. 2003.

[34] C. Dellarocas. Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior. In *Proceedings of the 2nd ACM conference on electronic commerce*, pages 150–157, Minneapolis, Minnesota, USA, 2000. ACM Press.

[35] Y. G. Desmedt. Threshold cryptography. *European Transaction on Telecommunications*, 5(4):449–457, July-August 1994.

[36] Z. Despotovic and K. Aberer. P2p reputation management: probabilistic estimation vs. social networks. *Computer Networks*, 50(4):485–500, 2006.

[37] G. Di Crescenzo, R. Ge, and G. R. Arce. Improved topology assumptions for threshold cryptography in mobile ad hoc networks. In *SASN '05: Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks*, pages 53–62, Alexandria, VA, USA, 2005. ACM Press.

[38] R. Di Pietro, L. V. Mancini, and G. Zanin. Efficient and adaptive threshold signatures for ad hoc networks. *Electronic Notes in Theoretical Computer Science (ENTCS)*, 171(1):93–105, 2007.

[39] S. Dobson, S. Denazis, A. Fernández, D. Gaïti, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt, and F. Zambonelli. A survey of autonomic communications. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 1(2):223–259, 2006.

[40] J. Douceur. The Sybil Attack. In *Proceedings of the 1st International Peer To Peer Systems Workshop (IPTPS 2002)*, pages 251–260, Cambridge, MA, USA, Mar. 2002. Springer.

[41] E. Fehr and S. Gaechter. Fairness and retaliation: The economics of reciprocitys. *Journal of Economic Perspectives*, 14:159–181, 2000.

[42] J. Feigenbaum and S. Shenker. Distributed Algorithmic Mechanism Design: Recent Results and Future Directions. In *In Proc. of Dial-M*, Atlanta, Georgia, USA, 2002. ACM.

[43] M. Feldman, K. Lai, I. Stoica, and J. Chuang. Robust incentive techniques for peer-to-peer networks. In *Proceedings of the 5th ACM conference on Electronic commerce (EC '04)*, pages 102–111, New York, NY, USA, 2004. ACM Press.

[44] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica. Free-riding and whitewashing in peer-to-peer systems. In *PINS '04: Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems*, pages 228–236, Portland, Oregon, USA, August 30 - September 3 2004. ACM Press.

[45] M. Felegyhazi, L. Buttyan, and J.-P. Hubaux. Nash equilibria of packet forwarding strategies in wireless ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(5):463–476, 2006.

[46] M. Felegyhazi and J.-P. Hubaux. Game Theory in Wireless Networks: A Tutorial. Technical Report LCA-REPORT-2006-002, EPFL, 2006.

[47] E. J. Friedman and P. Resnick. The social cost of cheap pseudonyms. *Journal of Economics & Management Strategy*, 10(2):173–199, 2001.

[48] C. Gamage, B. Gras, B. Crispo, and A. S. Tanenbaum. An Identity-based Ring Signature Scheme with Enhanced Privacy. In *Second International Conference on Security and Privacy in Communication Networks (SecureComm 2006)*, pages 1–5, Baltimore, MD, USA, Aug. 28 - Sep. 1 2006.

[49] A. Garg, R. Battiti, and R. Cascella. Reputation management: Experiments on the Robustness of ROCQ. In *Proceedings of the 7th International Symposium on Autonomous Decentralized Systems (First International Workshop on Autonomic Communication for Evolvable Next Generation Networks)*, pages 725–730, Chengdu, China, Apr. 2005.

[50] A. Garg, R. Battiti, and G. Costanzi. Dynamic Self-management of Autonomic Systems: The Reputation, Quality and Credibility (RQC) scheme. In *The 1st IFIP TC6 WG6.6 International Workshop on Autonomic Communication (WAC 2004) (LNCS 3457)*, pages 165–176, Berlin, Germany, Oct. 2004. Springer.

[51] A. Garg and R. Cascella. Reputation management for collaborative content distribution. In *Proceedings of the First International IEEE WoWMoM Workshop on Autonomic Communications and Computing, Taormina, Italy*, pages 547–552, Taormina, Sicily, Italy, June 13-16 2005. IEEE Computer Society.

[52] A. Garg, A. Montresor, and R. Battiti. Reputation Lending for Virtual Communities. In *Proceedings of the 22nd International Conference on Data Engineering Workshops (ICDEW'06)*, page 22, Atlanta, Georgia, USA, April 3-8 2006. IEEE Computer Society.

[53] R. Gibbons. *A Primer in Game Theory*. Prentice Hall, 1992.

[54] C. Gkantsidis and P. Rodriguez. Network Coding for Large Scale Content Distribution. In *24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005)*, volume 4(4), pages 2235–2245, Miami, FL, USA, March 13-17 2005.

[55] C. Gkantsidis and P. Rodriguez. Cooperative Security for Network Coding File Distribution. In *25th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2006)*, Barcelona, Spain, April 2006.

[56] P. Golle, K. Leyton-Brown, and I. Mironov. Incentives for sharing in peer-to-peer networks. In *Proceedings of the 3rd ACM conference on Electronic Commerce (EC '01)*, pages 264–267, Tampa, Florida, USA, 2001. ACM Press.

[57] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang. A Performance Study of BitTorrent-like Peer-to-Peer Systems. *IEEE Journal on Selected Areas in Communication (JSAC), Special Issue on Peer-to-Peer Communications and Applications*, 25(1), January 2007.

[58] M. Gupta and M. H. Ammar. Service differentiation in peer-to-peer networks utilizing reputations. In *5th COST264 International Workshop on Networked Group Communications (NGC 2003)*, volume 2816 of *Lecture Notes in Computer Science*, pages 70–82, Munich, Germany, September 16-19 2003. Springer.

[59] R. Gupta and A. K. Somani. Game Theory As A Tool To Strategize As Well As Predict Nodes Behavior In Peer-to-Peer Networks. In *Proceedings of the 11th International Conference on Parallel and Distributed Systems (ICPADS'05)*, pages 244–249, Washington, DC, USA, 2005. IEEE Computer Society.

[60] G. Hardin. The Tragedy of the Commons. *Science*, 162:1243–1248, 1968.

[61] E. Huang, J. Crowcroft, and I. Wassell. Rethinking incentives for mobile ad hoc networks. In *PINS '04: Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems*, pages 191–196, Portland, Oregon, USA, September 2004. ACM Press.

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

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

[64] W. Kellerer, Z. Despotovic, M. Michel, Q. Hofstatter, and S. Zols. Towards a Mobile Peer-to-Peer Service Platform. In *SAINT-W '07: Proceedings of the 2007 International Symposium on Applications and the Internet Workshops*, page 2, Hiroshima, Japan, January 15-19 2007. IEEE Computer Society.

[65] J. O. Kephart and D. M. Chess. The Vision of Autonomic Computing. *Computer, IEEE*, Volume 36, Issue 1:pages 41–50, January 2003.

[66] A. Kiayias and M. Yung. Group signatures with efficient concurrent join. In *Advances in Cryptology EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science (LNCS)*, pages 198–214, 2005.

[67] M. N. Krohn, M. J. Freedman, and D. Mazieres. On-the-Fly Verification of Rateless Erasure Codes for Efficient Content Distribution. In *IEEE Symposium on Security and Privacy*, Oakland, CA, USA, May 9-12 2004.

[68] C. Lampe and P. Resnick. Slash(dot) and burn: distributed moderation in a large online conversation space. In *Proceedings of the 2004 conference on Human factors in computing systems*, pages 543–550, Vienna, Austria, 2004. ACM Press.

[69] A. Legout, G. Urvoy-Keller, and P. Michiardi. Rarest first and choke algorithms are enough. In *IMC '06: Proceedings of the 6th ACM SIGCOMM on Internet measurement*, pages 203–216, Rio de Janeriro, Brazil, October 25-27 2006. ACM Press.

[70] J. Liang, R. Kumar, Y. Xi, and K. W. Ross. Pollution in file sharing systems. In *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005)*, volume 2, pages 1174–1185, Miami, FL, USA, March 13-17 2005. IEEE.

[71] X. Liu and L. Xiao. hirep: Hierarchical reputation management for peer-to-peer systems. In *ICPP '06: Proceedings of the 2006 International Conference on Parallel Processing*, pages 289–296, Columbus, Ohio, USA, August 14-18 2006. IEEE Computer Society.

[72] S. Marti and H. Garcia-Molina. Quantifying agent strategies under reputation. In *Fifth IEEE International Conference on Peer-to-Peer Computing (P2P '05)*, pages 97–105, Konstanz, Germany, August - September 2005. IEEE Computer Society.

[73] S. Marti and H. Garcia-Molina. Taxonomy of trust: categorizing p2p reputation systems. *Computer Networks*, 50(4):472–484, 2006.

[74] P. Michiardi and R. Molva. Core: a collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks. In *Proceedings of the IFIP TC6/TC11 6th Joint Working Conference on Communications and Multimedia Security*, pages 107–121, 2002.

[75] R. Morselli, J. Katz, and B. Bhattacharjee. A Game-Theoretic Framework for Analyzing Trust-Inference Protocols. In *Second Workshop on the Economics of Peer-to-Peer Systems (P2Pecon 2004)*, Cambridge, MA, USA, June 4-5 2004.

[76] N. Nisan and A. Ronen. Algorithmic Mechanism Design. *Games and Economic Behavior*, 35:166–196, 2001.

[77] M. Nowak and K. Sigmund. Evolution of indirect reciprocity. *Nature*, 437:1291–1298, October 2005.

[78] C. O'Riordan. Iterated Prisoner's Dilemma: A Review. Technical Report NUIG-IT-260601, Department of Information Technology, National University of Ireland, Galway, 2001.

[79] M. Piatek, T. Isdal, T. E. Anderson, A. Krishnamurthy, and A. Venkataramani. Do incentives build robustness in bittorrent? In *4th Symposium on Networked Systems Design and Implementation (NSDI)*, Cambridge, Massachusetts, USA, April 11-13 2007.

[80] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. In *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 161–172, San Diego, CA, USA, August 2001. ACM Press.

[81] P. Resnick and R. Zeckhauser. Trust among strangers in internet transactions: Empirical analysis of ebay's reputation system. the economics of the internet and e-commerce. In M. R. Baye, editor, *Advances in Applied Microeconomics*, volume 11, pages 127–157. Elsevier Science, 2002.

[82] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara. Reputation systems: Facilitating trust in internet interactions. *Communications of the ACM*, 43(12):45–48, 2000.

[83] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling churn in a DHT. In *In Proc. USENIX 2004 Annual Technical Conference*, pages 127–140, Boston, MA, USA, June 27 – July 2 2004.

[84] R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret: Theory and applications of ring signatures. In *ASIACRYPT '01: Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security*, pages 552–565, London, UK, 2001. Springer-Verlag.

[85] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 329–350, Heidelberg, Germany, Nov, 12-16 2001. Springer.

[86] S. Shin, J. Jung, and H. Balakrishnan. Malware prevalence in the kazaa file-sharing network. In *IMC '06: Proceedings of the 6th ACM SIGCOMM on Internet measurement*, pages 333–338, Rio de Janeriro, Brazil, October 25-27 2006. ACM Press.

[87] A. Singh and L. Liu. Trustme: Anonymous management of trust relationships in decentralized p2p systems. In *P2P '03: Proceedings of the 3rd International Conference on Peer-to-Peer Computing*, pages 142–149, Linkoping, Sweden, September 1-3 2003. IEEE Computer Society.

[88] E. Sit and R. Morris. Security considerations for peer-to-peer distributed hash tables. In *IPTPS '02: The First International Workshop on Peer-to-Peer Systems*, pages 261–269, Cambridge, MA, USA, March 7-8 2002. Springer-Verlag.

[89] V. Srivastava, J. Neel, A. B. Mackenzie, R. Menon, L. DaSilva, J. E. Hicks, J. H. Reed, and R. P. Gilles. Using game theory to analyze wireless ad hoc networks. *IEEE Communciations Surveys & Tutorials*, 7(4):46–56, 2005.

[90] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 ACM SIGCOMM Conference*, volume 31(4), pages 149–160, New York, NY, USA, August 2001. ACM Press.

[91] R. L. Trivers. The evolution of reciprocal altruism. *The Quarterly Review of Biology*, 46(1), March 1971.

[92] S. Čapkun, J.-P. Hubaux, and L. Buttyán. Mobility helps peer-to-peer security. *IEEE Transactions on Mobile Computing*, 5(1):43–51, January 2006.

[93] V. K. Wei and T. H. Yuen. (hierarchical identity-based) threshold ring signatures. Report 2006/193, Cryptology ePrint Archive, 2006.

[94] L. Xiong and L. Liu. PeerTrust: Supporting reputation-based trust in peer-to-peer communities. *IEEE Transactions on Data and Knowledge Engineering, Special Issue on Peer-to-Peer Based Data Management*, 16(7):843–857, July 2004.

[95] B. Yang and H. Garcia-Molina. Ppay: micropayments for peer-to-peer systems. In *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*, pages 300–310, Washington D.C., USA, October 27-30 2003. ACM Press.

[96] T. H. Yuen and V. K. Wei. Constant-size hierarchical identity-based signature/signcryption without random oracles. Report 2005/412, Cryptology ePrint Archive, 2005.

[97] G. Zacharia, A. Moukas, and P. Maes. Collaborative reputation mechanisms in electronic marketplaces. In *Proceedings of the Thirty-second Annual Hawaii International Conference on System Sciences-Volume 8*, pages 8026–8032. IEEE Computer Society, Jan. 1999.

[98] R. Zhou and K. Hwang. Gossip-based reputation aggregation for unstructured peer-to-peer networks. In *21th International Parallel and Distributed Processing Symposium (IPDPS 2007)*, pages 1–10, Long Beach, CA, USA, March 26-30 2007. IEEE.