N° d'ordre 2010-ISAL-0066 Année 2010

Thèse

Privacy Preserving Reputation Systems for Decentralized Environments

Présentée devant :
l'Institut National des Sciences Appliquées de Lyon

Pour obtenir :
Le grade de docteur

École doctorale :
Informatique et Mathématiques

Spécialité :
Informatique

Par :
Omar HASAN

Soutenue le 17 septembre 2010 devant le jury composé de :

Prof. Marc SHAPIRO, Université Paris 6, France ..... Président du jury
Prof. Ernesto DAMIANI, Université de Milan, Italie ..... Rapporteur
Prof. Michel HURFIN, INRIA Rennes, France ..... Rapporteur
Dr. Licia CAPRA, University College London, Royaume-Uni Examinatrice
Dr. David COQUIL, Université de Passau, Allemagne ..... Examinateur
Prof. Lionel BRUNIE, INSA de Lyon, France ..... Directeur de thèse
Prof. Jean-Marc PIERSON, U. Toulouse 3, France . Co-directeur de thèse

Abstract

Reputation systems are a key technology for making users accountable for their behavior in online communities. A reputation system computes the reputation of a user based on the feedback provided by the community. If the user draws negative feedback, it loses good reputation and may eventually be excluded from the network.

It has been observed that users in a reputation system often hesitate in providing negative feedback due to the fear of retaliation. A solution to this issue is privacy preserving reputation systems, which compute reputation such that the individual feedback of any user is not revealed. However, many existing privacy preserving reputation systems rely on centralized constructs and are thus not suitable for decentralized environments. The ones that are decentralized are either limited to specialized platforms (such as anonymous networks and trusted platforms), rely on trusted third parties, do not protect privacy under strict adversarial models, or are expensive in terms of resource utilization (for example, $O(n^3)$ messages, where n is the number of users providing feedback).

In this thesis, we present privacy preserving reputation protocols, that are decentralized, do not require specialized platforms nor trusted third parties, protect privacy under a range of adversarial models (semi-honest, non-disruptive malicious, disruptive malicious), and are more efficient than comparable protocols (the most expensive protocol requires $O(n) + O(log\ N)$ messages, where N is the total number of users). The techniques that we utilize include trust awareness, data perturbation, secret sharing, secure multi-party computation, additive homomorphic cryptosystems, and zero-knowledge proofs.

Another key innovation in our protocols is that an entity is able to quantify the probability of disclosure of its feedback. If the risk is unacceptable, the entity can abstain from submitting feedback. The protocols are thus able to offer up to perfect privacy, which has been previously assumed to be impossible in protocols for decentralized environments that compute reputation in an additive manner.

We also address some issues related to trust recommendation and propagation. In particular, we present a solution to the problem of subjectivity in trust recommendation. Experimental results indicate the effectiveness of the proposed strategies.

Acknowledgments

My first and foremost thanks go to my advisor, Prof. Lionel Brunie. I am very fortunate to have had Lionel's complete support, encouragement, and advice at every step throughout my PhD. One of the many memorable moments of my PhD was on my first day in Lyon, when I was very pleasantly surprised to find that Lionel himself had come to welcome me at the Perrache train station. I must also say *merci* to Lionel's family, Benedicte and kids, for their kind hospitality.

I am very grateful to my co-advisor, Prof. Jean-Marc Pierson, who invited me to INSA Lyon for my doctoral studies. Even though Jean-Marc moved to Toulouse, he continued to give me important advice and encouragement. From our long distance discussions under tight deadlines, to helping me with credit card troubles while we were in China, I always had Jean-Marc's support whenever I needed it.

I would like to thank Prof. Elisa Bertino, who has played an important role in the success of this thesis. My half-year long stay at her lab at Purdue University was the source of the idea for the topic of the thesis. Since then she has provided me valuable advice and direction.

I am immensely grateful to the *rapporteurs*, Prof. Ernesto Damiani and Prof. Michel Hurfin, and the *examinateurs*, Dr. Licia Capra, Dr. David Coquil, and Prof. Marc Shapiro. My thesis would not have been possible without their very generous participation as the members of the jury. They accepted to take significant amounts of time out of their schedules to review my thesis and to travel to Lyon for the defense. Their extremely valuable comments have provided me a lot of insight and many ideas for future work.

I wish to thank all members of the DRIM team and LIRIS, who also played a vital role in my PhD. The *permanents* provided excellent discussions and advice. My fellow *doctorants* provided wonderful company and lots of practical help. I would also like to thank all the secretaries and the administrative staff of LIRIS and INSA Lyon.

My success would of course not be possible without the support, the encouragement, and the prayers of all my family and friends. I had many fruitful discussions on the topics presented in this thesis with my brother, Ammar, who is an electrical engineer and a mathematician. No form of thanks would suffice for my parents. I entirely owe my success to their constant love, encouragement, and support.

Thank you everyone! May you all enjoy perfect privacy and excellent reputations!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Online communities that exist on the Internet and on ad-hoc networks present a variety of valuable opportunities for users. However, many such opportunities require interaction with strangers of unknown trustworthiness and thus entail significant risk. Some examples of the opportunities that a user may avail online by transacting with strangers and the associated risks are as follows:

- On e-commerce websites (such as eBay, Amazon), buyers are able to purchase a wide variety of items of interest, for example, electronic equipment, rare antiques, and excellent bargains. However, there is also the risk that the seller, represented by an anonymous pseudonym, turns out to be fraudulent. According to a survey on fraud in e-commerce [33], fraud accounted for a total loss of US$ 3.3 billion in the United States and Canada in 2009.

- In online social networks (such as Facebook, MySpace, Twitter), a member may befriend a user who claims to be a certain person. However, it is possible that the user is an imposter, hiding behind a digital personification of the claimed person. A recent case involved a woman in Missouri, USA who used a fake profile on MySpace to intimidate a teenage girl into committing suicide [127]. Apart from this extreme scenario, there have been cases where fake online persona have hijacked the identity of professionals and have succeeded in connecting to their real network of acquaintances [6]. This phenomenon may result in damages such as difficulties in employment, lost business contacts, etc.

- In peer-to-peer file sharing networks (such as BitTorrent), a user may download a useful file from a *seeder*. However, there is a risk that the file uploaded by the seeder is not useful at all, but fake content planted there for ulterior purposes, such as polluting the network or unsolicited dissemination.

- In a mobile ad-hoc network, a node may depend on a neighbor to relay its messages, however, the neighbor may be selfish and may drop those messages to conserve its resources.

Reputation is the general opinion of the community about the trustworthiness of an individual or an entity. A person who needs to interact with a stranger, often considers her reputation to determine the amount of trust that he can place in her. In the physical world, reputation comes from word of mouth, media coverage, physical infrastructure, etc. However, the reputation of a stranger is often difficult to observe in online communities, primarily due to their global scale, the cheap availability of anonymous identities, and the relative ease of acquiring high quality digital presence.

In recent years, reputation systems have gained popularity as a solution for securing distributed applications from misuse by dishonest entities. A reputation system computes the reputation scores of the entities in the system based on the feedback (quantified trust) provided by fellow entities. A reputation system makes an entity accountable for its behavior by creating the possibility of losing good reputation and eventual exclusion by the community. Reputation systems make certain that users are able to gauge the trustworthiness of an entity based on the history of its behavior. The expectation that people will consider one another's pasts in future interactions constrains their behavior in the present [112].

A popular reputation system is the eBay reputation system (`ebay.com`), which is used to discourage fraudulent activities in online auctions. After purchasing an item, a buyer can assign positive, neutral, or negative feedback to the seller depending on his level of satisfaction. The sum of the feedback provided by all buyers over a certain amount of time is considered as the reputation of the seller. The score is an indication of how likely the seller is to provide the service that he promises.

Reputation systems have also been proposed for: Rooting out fake profiles on social networks (these systems include Unvarnished [133], Duedil [44]); Defeating pollution in peer-to-peer file sharing networks (examples include Costa and Almeida [32], Yu [142], EigenTrust [80]); Discouraging selfish behavior in mobile ad-hoc networks (examples include Hu and Burmester [72], Buchegger et al. [21, 20]).

The reputation score of a target entity is a function of the feedback values provided by other entities. Thus an accurate reputation score is possible only if the feedback is accurate. However, it has been observed that the users of a reputation system may avoid providing honest feedback [111]. The reasons for such behavior include fear of retaliation from the target entity or mutual understanding that a feedback value would be reciprocated.

eBay originally allowed buyers and sellers to assign each other positive, neutral or negative feedback. A study [111] of eBay's reputation system revealed that there is a high correlation between buyer and seller feedback and over 99% of the feedback is positive. As discussed in [105], this could either imply that mutually satisfying transactions are in fact the norm or that the users are not

providing honest feedback due to the above cited reasons. The actual cause was obvious as the latter when eBay revised its reputation system [45] citing that "*... the [earlier] feedback system made some buyers reluctant to hold sellers accountable. For example, buyers fear retaliatory feedback from sellers if they leave a negative.*" Now sellers are not permitted to assign negative or neutral feedback to the buyers.

A more general solution to the problem of fear of retaliation for providing honest feedback is computing reputation scores in a privacy preserving manner. A privacy preserving reputation system operates such that the individual feedback of any entity is not revealed to other entities in the system. The implication of private feedback is that there are no consequences for the feedback provider and thus he is uninhibited to provide honest feedback.

Decentralized environments include peer-to-peer networks, ad-hoc networks, decentralized social networks etc. These environments are characterized by the absence of a trusted central authority. Several works have argued for the importance of decentralized reputation systems and proposed such systems (including, Quercia [110], Donato et al. [41], McNamara et al. [97]) .

A privacy preserving reputation system for centralized environments is fairly straightforward. Users submit their feedback to the trusted central authority who then computes and disseminates the reputation score while keeping the individual feedback private. However, preserving privacy in reputation systems for decentralized environments is challenging since there is no trusted central entity to rely on. Existing privacy preserving reputation systems either rely on centralized constructs (Androulaki et al. [9], Steinbrecher [126]), are limited to specialized hardware or networks (Nin et al. [101], Kinateder and Pearson [81]), do not protect privacy under stricter adversarial models (secure-sum [31], Gudes et al. [61]), or are expensive in terms of resource utilization (Pavlov et al. [105]).

In this thesis, our primary objective is to construct privacy preserving reputation protocols that are decentralized, do not require specialized platforms, protect privacy under standard adversarial models, and are efficient in terms of the number of messages exchanged and the bandwidth utilized.

More specifically, the challenge is to design a protocol that allows a querying agent to learn the correct reputation of a target agent. The reputation is an aggregate (for example, sum, mean) of the feedback values held privately by n source agents. Some of the requirements that the protocol must fulfill: 1) centralized entities, Trusted Third Parties (TTPs), and specialized platforms are not permitted; 2) all feedback values must remain private, that is, no agent is able to learn the feedback value of any other agent; 3) a source agent is unable to contribute feedback that lies outside the valid interval (even though the feedback is private); and 4) an agent who selectively drops messages or pre-maturely aborts the protocol is easily identified. We consider the following adversarial models: the semi-honest model, the non-disruptive malicious model, and the disruptive malicious model (the models are introduced in the next section and described more precisely in Section 3.4.1).

1.2 Contributions

In addition to an in depth survey of privacy preserving reputation systems (Section 2.3) and a framework that unifies the concepts of trust, reputation, and privacy in reputation systems (Chapter 3), this thesis makes the contributions discussed in the following subsections.

1.2.1 Protocols for the Semi-Honest Adversarial Model

The semi-honest adversarial model requires that all agents follow the protocol according to specification. However, the adversary passively attempts to learn the inputs of honest agents by using intermediate information received during the protocol and any other information that it can gain through other legitimate means.

A key innovation in our protocols is that agents take advantage of the trust in their fellow agents for their privacy. This trust awareness in the context of preserving privacy leads to protocols where privacy guarantees can be quantified before submitting the feedback. Moreover, this approach also results in protocols that are efficient.

Our first protocol (Round-Trip) builds upon the secure multi-party computation techniques of the secure-sum protocol [31]. Additional techniques utilized by the round-trip protocol include trust awareness, data perturbation, and a "round-trip" of messages instead of forwarding messages in a single direction. Round-trip provides security under the full semi-honest model, whereas secure-sum provides security only under a restricted model where agents are not allowed to collude.

An agent in the round-trip protocol is able to quantify the risk to its privacy as subjective probability from the amount of trust that it holds in the participants. Based on this information, it can either decide to contribute its private feedback or otherwise abstain. Through experiments on the large and real trust graph of Advogato.org, we demonstrate that even if only 40% of the source agents decide to participate, the protocol computes highly accurate reputation scores (as the mean of all feedback). Moreover, round-trip does not compromise on efficiency as it requires an exchange of $O(n)$ number of messages, where n is the number of agents in the protocol. This is the same complexity offered by the secure-sum protocol.

Pavlov et al. [105] argue that it is impossible to guarantee perfect privacy for an honest feedback provider in a protocol that computes reputation in an additive manner. However, the argument assumes that the set of n feedback providers is chosen deterministically by the adversary, with $n - 1$ dishonest agents, and that all feedback providers are obligated to submit feedback. We challenge this argument by allowing an honest agent to quantify its privacy guarantee beforehand and to abstain if the guarantee is not satisfactory.

Our second protocol (k-Shares) combines trust awareness with secret sharing. Shares of a private feedback value are generated such that it can be revealed only if all shares are known. Each feedback provider creates $k + 1$ shares

of its feedback value and distributes k shares among trustworthy agents, where $k \ll n-1$ and n is the number of feedback providers in the protocol. We hypothesize that it is possible for an agent to obtain a high enough privacy guarantee by distributing shares of its private feedback to only a maximum of k participants instead of all $n-1$ participants (as in comparable protocols, such as [105]). A clear advantage is linear communication complexity instead of quadratic. Our experimental results show that in the Advogato.org trust graph, the k-shares protocol will preserve the privacy of over 85% of instances of feedback providers with k set as low as 2. Raising k towards $n-1$ offers minimal improvement with the percentage rising to only approximately 87% at $k = n-1$ (which would be the case in the comparable protocol in [105] with a much higher communication complexity of over $O(n^2)$).

We develop an algorithm that takes any trust graph as input and determines the optimal value of k for running the k-Shares protocol in that graph. The optimal value of k for a trust graph is the value where increasing it gives no significant advantage in preserving the privacy of the potential source agents in the graph. A global view of the trust graph may not be available due to privacy concerns and decentralization, therefore we also develop an algorithm that operates on the subgraph known to a single feedback provider. The algorithm obtains the optimal value of k from that feedback provider's perspective.

1.2.2 Protocols for the Malicious Adversarial Models

Agents under a malicious model may deviate from the protocol as and when they deem necessary. To achieve their objectives, they may participate in extra-protocol activities, devise sophisticated strategies, and exhibit arbitrary behavior. A non-disruptive malicious adversary executes the malicious actions only if they lead to the disclosure of the inputs of honest agents. Whereas, a disruptive malicious adversary may act maliciously simply to disrupt the protocol.

We propose extensions that allow our protocols for the semi-honest model to provide security under the non-disruptive malicious adversarial model. One such extension is a Distributed Hash Table (DHT) to assign "source managers" to each agent. Source managers are fellow agents in the system that independently manage the list of feedback providers for an agent. The result is that an adversary is unable to prune the list of feedback providers to its advantage.

The protocol that we design for the disruptive malicious model is a version of our k-shares protocol, enhanced with the following cryptographic constructs: 1) an additive homomorphic cryptosystem (for example, the Paillier cryptosystem), and 2) zero knowledge proofs.

Additive homomorphic cryptosystems have the property that given only the ciphertexts of two numbers, the ciphertext of the sum of those two numbers can be computed. For example, given the encryptions $E(3)$ and $E(4)$, it is possible to determine $E(7)$ without ever knowing the numbers 3 and 4. Zero-knowledge proofs allow a prover to convince a verifier that a statement is correct, without revealing any additional information. For example, a prover may prove that a secret number (for example, 7) lies in a given interval (for example, $[0, 10]$)

without revealing the number.

A majority of the protocols (such as Androulaki et al. [9], Steinbrecher [126], Kinateder and Pearson [81]) for the disruptive malicious model rely on centralized constructs or specialized platforms. Our protocol for the disruptive malicious model is decentralized and general purpose yet efficient. The number of messages exchanged is $O(n) + O(log\ N)$, where n and N are the number of feedback providers in the protocol and the total number of agents in the system respectively. This is a significant improvement over the comparable protocol in [105], which requires $O(n^3)$ messages.

1.2.3 Trust Recommendation and Propagation

We also address some issues in trust recommendation and propagation. These techniques are related to reputation as they also aid users in establishing trust in unknown entities.

We argue that when a trust value is recommended by one user to another, it may lose its real meaning due to subjectivity. Let's say that Alice tells Bob that she has 0.7 (in $[0, 1]$) trust in Carol. A value of 0.7 might mean average trustworthiness for Alice, whereas Bob could have a different interpretation of the same value (for example, high trustworthiness). We present a solution based on the notion of percentiles for the elimination of subjectivity from trust recommendation. Experiments on simulated trust graphs with high subjectivity show promising results.

Trust propagation, that is acquiring trust in a distant entity based on the existence of a connecting trust path, is a fundamental technique in trust management. Through analysis of the trust graph of Advogato.org, we demonstrate that a substantial positive linear correlation exists between the amount of trust acquired through direct interaction, and trust acquired through the technique of trust propagation. This result provides validation for the trust propagation technique.

1.3 Thesis Outline

A brief outline of the remainder of the thesis is as follows:

Chapter 2 – Trust, Reputation, and Privacy. Background and a survey of the literature in the areas of trust, reputation, and privacy preserving reputation systems.

Chapter 3 – General Framework. Establishes a formal framework for the thesis. The framework formalizes the concepts of trust, reputation, and privacy in an integrated manner.

Chapter 4 – Protocols for the Semi-Honest Adversarial Model. We present several protocols for computing reputation in a privacy preserving

manner under the semi-honest adversarial model. The protocols are analyzed for correctness, privacy, and efficiency. Each protocol is evaluated using a real dataset. We compare the protocols with prior literature.

Chapter 5 – Protocols for the Malicious Adversarial Models. Our reputation protocols that provide privacy under the non-disruptive as well as disruptive malicious models are presented in this chapter. The protocol for the disruptive malicious model utilizes cryptographic tools such as an additive homomorphic cryptosystem and various zero-knowledge proofs. We analyze the protocols and give comparisons with existing solutions.

Chapter 6 – Trust Recommendation and Propagation. In this chapter we tackle issues in trust recommendation and propagation. We develop and evaluate a solution to the problem of subjectivity in trust recommendation. We also analyze whether iterative multiplication is a suitable strategy for trust propagation. The issue of privacy as it relates to trust recommendation and propagation is also discussed.

Chapter 7 – Conclusion. Conclusions and future directions are presented in this final chapter.

Chapter 2

Trust, Reputation, and Privacy

In this chapter, we provide background on the principal themes of this thesis, which are trust, reputation, and privacy in reputation systems.

We begin with a look at the significance of trust in information security. In our discussion on trust, we review some of the available methods for establishing trust. These methods include direct interaction, trust recommendation and propagation, trust negotiation, and reputation.

Reputation is the opinion of the community about the trustworthiness of an entity. Reputation systems compute reputation scores from the feedback provided by the community and aid entities in establishing trust in each other. We discuss the characteristics that define a reputation system, the various feedback aggregation models, as well as the challenges in designing a reputation system. One of the key challenges in reputation systems is the lack of privacy. This issue often prevents users from providing honest feedback due to the fear of retaliation.

In this thesis, we focus on solutions to the issue of lack of privacy in reputation systems. We present an in-depth overview of the various existing reputation systems that aim to preserve privacy. We conclude with a comparison of these existing systems and a justification for why we need novel solutions.

2.1 Trust

In recent years, trust has garnered considerable interest in the computer science community as a building block for solutions to various information security issues, such as authentication, access control, and service provisioning.

Systems such as X.509 [2] and PGP [53] utilize trust to authenticate the identity of entities even if they are previously unknown. They employ constructs such as *trusted Certificate Authority (CA)*, *web of trust* and *trust paths* to accomplish this task. The underlying concept is that the claimed identity of an

entity is considered genuine if a trustworthy third-party certifies that identity.

Traditional access control mechanisms such as Identity Based Access Control (IBAC) and Role Based Access Control (RBAC) are not always suitable for environments where nodes are mostly anonymous (such as peer-to-peer file sharing networks, ad-hoc networks) and thus cannot be granted access according to identities and roles. Trust based access control mechanisms (such as [132, 143, 67]) provide alternative solutions that exploit the level of trust of the provider in other entities to govern access.

In networked communities, such as those on the Internet and ad-hoc networks, users frequently have the opportunity to obtain beneficial services from complete strangers. For example: on an e-commerce website, a buyer could purchase an item from an unknown seller; in a peer-to-peer file sharing network, a peer could download a file from a seeder; in an ad-hoc network, a node could use the services of another node to relay a message. However, in all of these cases, there is considerable risk that the service provider would behave maliciously. Techniques for establishing trust (such as those discussed in Section 2.1.3) help consumers determine whether a potential service provider would deliver the promised services.

In the following sections, we take a look at the various definitions of trust (Section 2.1.1), characteristics of trust (Section 2.1.2), and several techniques for establishing trust (Section 2.1.3).

2.1.1 Defining Trust

There has been extensive research on the concept of trust in many different domains, such as sociology, philosophy, social psychology, economics, and computer science. Therefore, a number of definitions of trust have been proposed in the literature with different perspectives. According to Marsh [90, page 20], a common element that links all studies on trust is the assumption of the presence of a society. In this section we present some of the influential definitions of trust that appear in the literature.

One of the earlier notable definitions of trust is formulated by social psychologist **Morton Deutsch** [36]. The definition states that when:

> 1. "the individual is confronted with an ambiguous path, a path that can lead to an event perceived to be beneficial ($Va+$) or to an event perceived to be harmful ($Va-$);
> 2. he perceives that the occurrence of $Va+$ or $Va-$ is contingent on the behavior of another person; and
> 3. he perceives the strength of $Va-$ to be greater than the strength of $Va+$.
>
> If he chooses to take an ambiguous path with such properties, I shall say he makes a trusting choice; if he chooses not to take the path, he makes a distrustful choice."

We interpret Deutsch as follows: Trust and distrust are discrete, distinct choices that arise when an individual must rely on another person to gain a benefit. However, there is also a possibility that relying on that person may actually lead to harm instead of benefit. The individual trusts that person if he chooses to rely on him, and distrusts him if he chooses otherwise.

In [52], sociologist **Diego Gambetta** proposes the following definition of trust:

> Trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent assesses that another agent or group of agents will perform a particular action, both before he can monitor such action (or independently of his capacity ever to be able to monitor it) and in a context in which it affects his own action.

This is one of the seminal definitions that describe trust as a quantifiable construct. Gambetta observes that trust is an agent's degree of belief (the level of subjective probability) that another entity will perform an expected action. An additional important aspect of this definition is the recognition that trust is contextual.

McKnight et al. [95], a prominent work in the field of trust in information systems, regards trust as a multidimensional concept composed of two constructs: trusting *intention*, and trusting *beliefs*. Trusting intention means that one is willing to depend on the other person in a given situation, and trusting beliefs imply that one believes the other person to be *competent*, *benevolent*, and having *integrity*. Competence is the ability of the trustee to do what the truster needs, benevolence is the trustee's caring and motivation to act in the truster's interests, and integrity is the trustee's honesty and promise keeping.

Grandison and Sloman [59] discuss trust from a computer science perspective. They consider trust to be a composition of many different attributes such as reliability, dependability, honesty, truthfulness, security, competence, and timeliness. Their definition of trust is as follows:

> The firm belief in the competence of an entity to act dependably, securely and reliably within a specified context (assuming dependability covers reliability and timeliness).

Jøsang et al.'s survey on trust and reputation systems [77] differentiates between two types of trust: reliability trust, and decision trust. Reliability trust, which is inspired by Gambetta's definition, is stated as:

> Trust (reliability trust) is the subjective probability by which an individual, A, expects that another individual, B, performs a given action on which its welfare depends.

This definition stresses on the *dependence* on a trustee, and the *reliability* (probability) of the trustee. However, Jøsang et al. concur with Falcone and

Castelfranchi [47], and McKnight and Chervany [94] in recognizing that having high reliability trust in a person is not necessarily enough to decide to enter into a situation of dependence on that person. For example, if the stakes are too high, then even a small probability of failure may lead an individual to decide to not depend on a potential trustee. Thus, trust could be viewed as the decision to depend or not depend on another entity. Jøsang et al. summarize *decision trust* as:

> Trust (decision trust) is the extent to which one party is willing to depend on something or somebody in a given situation with a feeling of relative security, even though negative consequences are possible.

In this thesis, we subscribe primarily to the definition of trust by sociologist Diego Gambetta [52]. The reason for this choice is the ability to quantify trust as probability, which consequently makes it possible to quantify the security guarantees of the protocols that we build.

2.1.2 Characteristics of Trust

From Gambetta's definition, we infer that trust has the following characteristics:

Binary-Relational and Directional. According to the definition, "Trust ... is a particular level of the subjective probability with which ***an agent*** assesses that ***another agent or group of agents*** will perform a particular action ...". From this excerpt, it is evident that trust is a relationship between two entities. Moreover, it is also clear that trust is directional. The first entity is an agent who has trust in a second entity which may be another agent or a group of agents.

Contextual. As given in the definition, "Trust ... is a particular level of the subjective probability with which an agent assesses that another agent or group of agents will perform ***a particular action*** ... ". We infer that trust is in the context of "a particular action" that the second entity may perform.

Quantifiable as Subjective Probability. "Trust ... is ***a particular level of the subjective probability*** with which an agent assesses that another agent or group of agents will perform a particular action, ***both before he can monitor such action (or independently of his capacity ever to be able to monitor it)*** ...". From this excerpt of the definition, we deduce that trust is quantifiable as subjective probability.

Subjective probability is the degree of the personal belief of an individual that an event will occur. The alternate approach to defining probability is in terms of the frequency of occurrence of observable physical events. However, subjective probability does not require observation of any past events. An individual may form his personal belief regarding the likelihood of the occurrence of an event based on past experiences or lack thereof.

Another important characteristic of subjective probability is that different individuals may hold different personal beliefs regarding the occurrence of the same event thus leading to different individual subjective probabilities. An example of subjective probability is the degree of the personal belief of an online shopper that an online vendor will deliver the promised product even though there have been no previous transactions between the two. We refer the reader to [121] for a seminal discussion on subjective probability.

We discuss below some other characteristics of trust which are not evident from Gambetta's definition. We provide examples to support their validity as characteristics of trust. These characteristics have been previously identified by several authors (such as [26]).

Non-Reflexive. An agent may or may not trust herself. For example, a patient Alice may trust her doctor to prescribe her the correct medicine, whereas she might not trust herself to do so.

Asymmetric. If an agent Alice trusts an agent Bob, then Bob may or may not trust Alice. For example, in the context of car repair, a car owner Alice may trust her mechanic Bob, however Bob may not necessarily trust Alice.

Non-Transitive. If an agent Alice trusts an agent Bob who in turn trusts an agent Carol, then Alice may or may not trust Carol. For example, an email server A might trust an email server B to not send spam. If B trusts an email server C in the same context, then A may or may not trust C depending on various factors such as its strength of trust in B, the availability of additional evidence, etc.

Dynamic. Trust may change with time. For example, let's say that an online shopper Alice has so far had good experiences with an online vendor Bob and therefore she has high trust in him. However, if her latest transaction with Bob is less than satisfactory, then her trust in Bob is likely to decrease instead of staying constant.

2.1.3 Establishing Trust

There are a number of techniques that enable establishing trust in unknown entities:

Direct Interaction

The primitive method of establishing trust in an unknown entity is to directly interact with it and observe its behavior in the desired context. However, this method requires that the entity be trusted at least once without any prior background on that entity. This approach is perhaps suitable for low-risk transactions and in situations when no other recourse is available. However, when

reliance on an unknown entity may lead to substantial damage, the other approaches for trust establishment are clearly preferable, since they allow the truster to base his trust on some prior knowledge provided by others.

McKnight et al. [96] introduce the notion of *initial trust*, which is described as the trust in an unfamiliar trustee — a relationship in which the actors do not yet have credible information about, or affective bonds with each other [17].

Trust Recommendation and Propagation

Establishing trust in an unknown entity through trust recommendation and propagation takes advantage of the possible transitivity of trust. Let's say that Alice wishes to establish trust in an unknown individual, Carol. If another individual Bob trusts Carol then he could give a recommendation to Alice about Carol's trustworthiness. Taking Bob's trust recommendation and her own trust in Bob into account, Alice may establish a trust relationship with Carol. Thus a transitive path of trust that leads from Alice to Bob to Carol, enables Alice to develop trust in Carol. If Alice wishes to establish trust in Carol through Bob's recommendation, we say that Bob's trust in Carol has propagated to Alice.

Guha et al. [62] term the above described one-step propagation as *atomic propagation*. The term stems from the observation that the conclusion is reached based on a single argument, rather than a possibly lengthy chain of arguments. Guha et al. identify four types of atomic propagations: *direct propagation*, *co-citation*, *transpose trust*, and *trust coupling*. We briefly elaborate each of these types of atomic trust propagation:

Direct Propagation. The example given in the first paragraph represents direct propagation. If i trusts j, and j trusts k, then a direct propagation allows us to infer that i trusts k. Guha et al. refer to this particular atomic propagation as direct propagation since the trust propagates directly along an edge.

Co-Citation. Let's consider that i_1 trusts j_1 and j_2, and i_2 trusts j_2. Under co-citation, it is concluded that i_2 also trusts j_1.

Transpose Trust. In transpose trust, i's trust in j causes j to develop some level of trust towards i. Let's say that i trusts j, then transpose trust implies that j should also trust i.

Trust Coupling. Let's suppose that i and j both trust k, then trust coupling leads us to infer that i and j should trust each other since they both trust k.

Iterative propagation builds upon multiple atomic propagations to help establish trust in an unknown entity. Let's extend the example presented in the first paragraph: Alice trusts Bob and Bob trusts Carol. We further assume that Carol trusts Dave. Alice may establish trust in Dave as a result of the following two atomic propagations: 1) the first atomic propagation builds Bob's direct trust in Dave, and 2) now since Bob trusts Dave, Alice can establish

trust in Dave through a second atomic propagation. This sequence of atomic propagations is referred to as iterative propagation.

Trust Negotiation

Trust negotiation is an approach that can enable strangers to electronically share sensitive data and services. Trust negotiation establishes trust between entities based not on their identities but their properties. For example, in the case of an individual, the properties that may be considered include their place of employment, age, membership in certain organizations etc. With trust negotiation, the trust between two entities is acquired through iterative requests for credentials and their disclosure.

An example from [16]: *CARS* is an online car rental agency, which has an agreement with a company called *CORRIER* to provide rental vehicles free of charge to their employees, provided that they prove their employment status (which also implies that they are authorized to drive). Other customers (who are not employees of *CORRIER*) can rent a vehicle by showing a valid driving license and by providing a credit card for payment. Thus, *CARS* establishes trust in customers to be legitimate drivers through the exchange of multiple possible credentials.

```
Customer: Request a vehicle
CARS: Show digital employment ID from CORRIER
Customer: Not available
CARS: Show digital driving license
Customer: Digital driving license
CARS: Provide digital credit card
Customer: Digital credit card
CARS: Vehicle granted (vehicle info, pickup info, etc.)
```

Two fundamental building blocks for trust negotiation are digital credentials and disclosure policies. A disclosure policy states the conditions that must be satisfied by a requesting party to gain access to a resource. The conditions are constraints against the properties of the credentials to be provided by the interacting parties.

Digital credentials contain sensitive information, therefore it is desired that the least amount of information be disclosed in the process of a trust negotiation. The disclosure of credentials is governed by the trust negotiation strategy of a party. A simple trust negotiation strategy is the *eager* strategy, in which parties send each other all unlocked credentials until the negotiation succeeds or until no new unlocked credentials are available and thus the negotiation halts.

Trust-X (Bertino et al. [16]) and TrustBuilder (Winslett et al. [139]) are two well-known trust negotiation systems. A novel aspect of the Trust-X policy language X-TNL is the support for special certificates, called trust tickets. Trust tickets are issued on successfully completing a negotiation and can speed up subsequent negotiations between the two parties. TrustBuilder defines a

strategy independent protocol, which allows two parties to negotiate trust even if they are using different trust negotiation strategies.

Privacy is a significant concern in trust negotiation. One of the issues is that once a credential has been disclosed to a party, there is no guarantee that it would not leak that credential to a third party.

Reputation

Reputation is the general opinion of the community about the trustworthiness of an individual or an entity. A person who needs to interact with a stranger, may analyze her reputation to determine the amount of trust that he can place in her. In the physical world, reputation often comes from word of mouth, media coverage, physical infrastructure, etc. However, the reputation of a stranger is often difficult to observe in online communities, primarily due to their global scale, the cheap availability of anonymous identities, and the relative ease of acquiring high quality digital presence.

A reputation system computes the reputation of an entity based on the feedback (quantified trust) provided by fellow entities. Reputation systems make certain that users are able to gauge the trustworthiness of an entity based on the history of its behavior. The expectation that people will consider one another's pasts in future interactions constrains their behavior in the present [112]. Political scientist Robert Axelrod calls this the "shadow of the future" [112, 12].

Using reputation systems for establishing trust is a principal focus of this thesis. Section 2.2 is dedicated to a detailed overview of reputation.

2.2 Reputation

Hoffman et al. [69] provide the following pertinent description of reputation:

> In general, reputation is the opinion of the public toward a person, a group of people, or an organization. In the context of collaborative applications such as peer-to-peer systems, reputation represents the opinions nodes in the system have about their peers. Reputation allows parties to build trust, or the degree to which one party has confidence in another within the context of a given purpose or decision. By harnessing the community knowledge in the form of feedback, reputation-based trust systems help participants decide who to trust, encourage trustworthy behavior, and deter dishonest participation by providing a means through which reputation and ultimately trust can be quantified and disseminated.

In recent years, reputation systems have become one of the most ubiquitous means of acquiring trust in strangers online (consider the reputation systems in use by eBay.com, Amazon.com, Advogato.org., etc.).

In the following sections, we describe the characteristics that define a reputation system (Section 2.2.1), feedback aggregation models (Section 2.2.2), and the challenges in reputation system design (Section 2.2.3).

2.2.1 Characteristics of Reputation Systems

Network Architecture

The network architecture of a reputation system is one of the key factors in determining how the following activities are conducted:

- Feedback collection
- Feedback aggregation (reputation computation)
- Reputation dissemination

The two common network architectures are: centralized and decentralized.

Centralized Reputation Systems. Centralized reputation systems are characterized by the existence of a trusted central authority. The central authority receives feedback from users, aggregates it to compute the reputation, and disseminates the reputation scores.

One of the benefits of a centralized solution is that it is straightforward to implement. Moreover, a centralized reputation system is often less vulnerable to certain attacks, such as the sybil attack (Section 2.2.3), since the central authority can monitor and correlate all activities in the reputation system. Additionally, the central authority is universally trusted, therefore users can be assured that the feedback collection, aggregation, and dissemination are being done correctly.

Unfortunately, the requirement of universal trustworthiness of the central authority is also a liability. If the central authority fails or becomes compromised, then the whole reputation system crashes. Thus the central authority is a single point of failure and a high-value target for attackers. As with any other centralized system, another major disadvantage of centralized reputation systems is that they are very expensive to deploy and maintain, particularly for large numbers of users. Centralized reputation systems are also unable to cater for decentralized environments such as peer-to-peer networks, ad-hoc networks, decentralized social networks, etc.

Examples of centralized reputation systems include ebay.com, epinions.com, amazon.com, advogato.org, and PageRank [103].

Decentralized Reputation Systems. Decentralized environments are characterized by the absence of a central authority. Advantages of such networks include: lack of a single point of failure, no need to deploy and maintain an expensive central authority, a more democratic environment, scalability, etc. Examples of decentralized environments include peer-to-peer networks, ad-hoc networks (such as Mobile Ad-Hoc Networks - MANETs, and Vehicular Ad-Hoc Networks - VANETs), decentralized social networks (such as FOAF), etc.

Decentralized reputation systems are suitable for decentralized environments as they do not assume the presence of a central entity. In decentralized reputation systems, a central location for submitting and aggregating feedback, and disseminating reputation does not exist. Feedback is commonly stored locally by the node who generates it, for example in response to his experiences with another party. Computing reputation of an entity in the system requires finding all or a portion of the nodes who carry feedback about that entity. Once the feedback providers have been located, the aggregation may be done at a single location after receiving all feedback, or a more sophisticated protocol may be employed to aggregate the feedback in a distributed manner.

Examples of decentralized reputation systems include Damiani et al. [35], Gupta et al. [63], EigenTrust [80], and PowerTrust [144].

Other Characteristics

Other characteristics that define a reputation system include: *reputation visibility*, *reputation durability*, *feedback durability*, and the *feedback aggregation model*.

The visibility of a reputation score may be global or local. Global visibility implies that all nodes in the system view the same reputation score of a certain entity. Whereas with local visibility, the reputation score available to a subset of the nodes may be different than elsewhere in the system. Local visibility is generally a concern in decentralized reputation systems, where a different subset of feedback providers may be included for computing the reputation of an entity at different instances.

Reputation durability refers to the transience of a reputation score. Once a reputation score is computed, it may be stored permanently for subsequent access by nodes through a simple retrieval operation. Recalculation of the score is mandated only when new feedback becomes available. Alternatively, the reputation score may be transient and re-computed every time a node wishes to learn the score. The latter approach requires repeated computation of the reputation, however, it does not require storage of the scores by a trustworthy entity.

Feedback durability refers to the lifetime of a feedback value. A feedback value may remain valid for an indefinite period of time or it may be considered obsolete with the passage of time. An obsolete feedback value may be entirely excluded from the reputation computation or its significance may be discounted.

A feedback aggregation model is required to compute reputation. These models include: summation and mean, flow network, Markov chain, and Bayesian. Section 2.2.2 is dedicated to the discussion of these models.

2.2.2 Feedback Aggregation Models

There are a number of models for aggregating feedback to obtain reputation scores. We recapitulate some of the common models below. An excellent survey

on these methods (also called reputation computation engines) is provided by Jøsang et al. [77].

Sum and Mean Model

One of the most common methods of aggregating feedback to obtain the reputation score is the simple summation of the feedback. The eBay reputation system (ebay.com) allows users to give positive (+1), neutral (0), or negative (−1) feedback. The reputation is computed as the sum of the feedback provided over a certain period of time. The higher the sum of the feedback, the better is the reputation of the user. The advantage of this approach is that it is intuitive and easy to understand.

Another related method is to compute the reputation score of an entity as the mean of the feedback values. Reputation represented as mean has the benefit of being normalized and thus the reputation of different entities may be compared objectively.

Many practical reputation systems use summation or mean as the preferred methods for computing reputation. Examples include ebay.com, epinions.com, and amazon.com.

Flow Network Model

A flow network is a weighted directed graph in which each edge is characterized by the capacity of flow that it can carry. A node in the network may receive and send flow through incoming and outgoing edges respectively. However, the amount of flow into a node must be equal to the amount of flow out of the node. This constraint does not apply to two special nodes called the source and the sink. The source can only have outgoing edges and its flow output is unlimited. Conversely, the sink node may only have incoming edges and may receive an unlimited amount of flow.

The Advogato (advogato.org) [87] and the Appleseed [145] reputation systems are constructed on the concept of flow networks. The feedback is considered as the flow in the network of nodes. The reputation of a node is computed as a function of the flow (the feedback) that the node receives. A salient characteristic of such reputation systems is that a node may only assign the amount of feedback that it has received. If the amount of flow available in the network is regulated by a few trustworthy nodes adjacent to the source, it becomes significantly challenging for malicious nodes to mount certain types of attacks. For example, it does not help an attacker to create multiple pseudonyms in the system to mount a sybil attack (discussed in Section 2.2.3), since the sum of the flow available to the pseudonyms remains constant.

Markov Chain Model

A Markov chain may be viewed as a weighted directed graph in which nodes represent states and the edges represent transitions between those states. The

weight of an edge expresses the probability of the corresponding transition. Each transition in a Markov chain is independent of any past states and transitions.

Several successful reputation systems [80, 103] have the Markov chain theory as their foundation. We describe the basic principle as follows: Let's say that feedback is the probability of transition from one node to another, then the reputation of a node may be considered as the probability of arriving at that node by following random transitions.

Google's PageRank algorithm [103] may be considered as a reputation system for web pages. The World Wide Web is defined as a Markov chain in which the transition probability from each page p_i to its k_i linked (outbound) pages is $\frac{\alpha}{k_i} + \frac{1-\alpha}{N}$, and $\frac{1-\alpha}{N}$ for all other pages on the web that page p_i does not have links to. α is a constant with a heuristic value of 0.85, and N is the total number of known web pages. The PageRank (or reputation) of a web page p_i is given as:

$$PR(p_i) = \frac{1-\alpha}{N} + \alpha \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{k_j} \tag{2.1}$$

where $p_1, p_2, ..., p_N$ are the pages on the web, $M(p_i)$ is the set of pages that link to p_i, k_j is the number of outbound links on page p_j, and N is the total number of known web pages.

EigenTrust [80] and PowerTrust [144] are other well-known reputation systems that draw on the Markov chain theory. A distinctive characteristic of EigenTrust is that it implements a Markov chain based reputation system for decentralized environments. This makes it suitable for applications such as peer-to-peer file sharing.

Bayesian Model

The beta distribution is a statistical distribution characterized by two free parameters, α and β. The domain of the distribution is $[0, 1]$. The probability function $P(x)$ and the distribution function $D(x)$ are defined as follows:

$$P(x) = \frac{(1-x)^{\beta-1} x^{\alpha-1}}{B(\alpha, \beta)} \tag{2.2}$$

$$= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} (1-x)^{\beta-1} x^{\alpha-1} \tag{2.3}$$

$$D(x) = I(x; a, b) \tag{2.4}$$

where Γ is the gamma function, $B(a, b)$ is the beta function, $I(x; a, b)$ is the regularized beta function, and $\alpha, \beta > 0$. The beta distribution is normalized due to the fact that:

$$\int_0^1 P(x)dx = 1 \tag{2.5}$$

In Bayesian statistics, the beta distribution is used as a prior distribution for binomial proportions. The beta distribution is one of the preferred distributions for describing the variety of opinion across people.

In a Bayesian reputation system, the reputation score is generally represented by a beta distribution, where the two free parameters α and β correspond to the number of positive and negative feedback respectively. The reputation score is computed by statistically updating the given beta distribution.

Bayesian reputation systems provide a sound mathematical basis for computing reputation scores. However, in contrast to other reputation models (for example, summation and mean), the Bayesian reputation model renders the reputation scores less intuitive. Bayesian reputation systems include [64, 137, 76].

2.2.3 Challenges in Reputation System Design

In this section, we discuss some of the open issues in reputation systems. An excellent discussion of possible attacks on reputation systems is also provided by Hoffman et al. [69]. Additionally, the reader may consult the survey by Jøsang et al. [77] for an overview of the problems faced by reputation systems.

Lack of Privacy

An accurate reputation score is possible only if the feedback is accurate. However, it has been observed that the users of a reputation system may avoid providing honest feedback [111]. The reasons for such behavior include fear of retaliation from the target entity or mutual understanding that a feedback value would be reciprocated.

A solution to the problem of fear of retaliation is computing reputation scores in a privacy preserving manner. A privacy preserving protocol for computing reputation scores does not reveal the individual feedback of any entity. Private feedback ensures that there are no consequences for the feedback provider and thus he is uninhibited to provide honest feedback.

Slandering is the act of sabotaging an honest user's reputation by assigning them unwarranted low feedback. A tradeoff of private feedback is that it creates the opportunity for slandering without consequences. However, we draw attention to the processes of voting and election, where the privacy of the voters is often guaranteed to allow them complete freedom of opinion. Since feedback providers in reputation systems are similarly entitled to personal opinion, it can be argued that their privacy should also be preserved. Slandering is most effective when it is carried out by a collusion of users. An important challenge to be addressed by future work (discussed in Section 7.2) is the detection of collusions in privacy preserving reputation systems.

In Section 2.3, we give an in depth discussion of some existing systems that compute reputation in a privacy preserving manner. The primary contribution of this thesis is the proposal of several novel privacy preserving reputation protocols.

Sybil Attack

The sybil attack [42] on a reputation system operates as follows: An attacker creates multiple identities in the system in order to gain an unfair advantage over honest users who own a single identity. The attacker may use its multiple identities to mount attacks including self-promotion, slandering, and ballot stuffing.

A reputation system is vulnerable to sybil attacks when it has one or more of the following characteristics:

- The reputation system does not require users to submit their true identities when creating a pseudonym in the system.
- There is no central authority to verify the true identity behind a pseudonym. Peer-to-peer reputation systems are thus particularly prone to sybil attacks.
- The reputation system accepts new pseudonyms that have a weak or no chain of trust linking them to trustworthy entities in the system.
- Creating a new pseudonym requires low resources.
- The reputation system treats all pseudonyms equally.

The eBay reputation system fights sybil attacks by differentiating between users who authenticate their true identities to the system (through a bank account, or credit card, etc.) and those who don't. The authenticated users get more privileges in the system. The central eBay server allows only one pseudonym per authenticated identity.

The Advogato [87] and Appleseed [145] reputation systems prevent this attack by reducing the influence of pseudonyms created by a single entity. Since an entity has a limited amount of flow received from existing entities, it does not help to create new pseudonyms and distribute that flow among them. The total influence of the entity remains the same.

Yu et al. [141] propose an approach based on social networks to detect sybil attacks. The algorithm operates by ensuring that the size of the cut between the set of known honest nodes and the set of potential attackers remains small.

Self-Promotion, Ballot Stuffing

Self-promotion is the act of raising one's own reputation through unfair means. Self-promotion may be carried out by a user individually or in collusion with other members of the system. When a reputation score is transmitted over insecure channels, an attacker may intercept the score and augment it to exhibit better reputation. Another self-promotion attack is possible in systems (such as eBay) where users may assign each other additional feedback after every transaction. Two users may repeatedly transact with each other, and after each transaction assign each other positive feedback. This attack is also known

as ballot stuffing, which implies that a user submits more feedback than he is entitled to. Another scenario is that a user creates multiple identities in the system (the sybil attack), and uses those fake multiple identities for self-promotion.

Prevention strategies for self-promotion include:

- Dissemination of reputation scores over secure channels.
- Permitting a user to assign only one feedback to another user at any given time.
- Placing a cost (financial or some other limited resource) on each transaction, making fake repeated transactions expensive.
- Preventing a user from controlling multiple identities (the sybil attack).
- Identifying and breaking up cliques who have a high number of transactions among themselves.

The strategy employed by the reputation systems of many online auction and e-commerce websites (for example, eBay, Amazon), is to charge the seller a fee for each transaction. Thus, repeated fake transactions for the purpose of accumulating feedback becomes costly.

Slandering

As introduced earlier, slandering is the act of sabotaging an honest user's reputation by assigning them unwarranted low feedback. Motivation for such an attack may include retaliation, reducing a competitor's reputation, or malicious disruption of services. A slandering attack is particularly detrimental to the target user in applications that are sensitive to the presence of even a small amount of low feedback, such as high-value monetary systems. The slandering attack may be carried out by an individual or a coalition of entities. The attack is clearly more effective in the latter case, especially if the size of the coalition is large. The slandering attack is hard to detect and prevent, since it is a legitimate right of a user to assign low feedback.

Some counter strategies include:

- Third party arbitration in case of negative feedback.
- Identification of users or cliques who provide unusually high proportion of low feedback.
- Allowing a user to explain its position to other members if it is given negative feedback.
- Preventing sybil attacks.

The reputation system by Belenkiy et al. [14] (discussed in Section 2.3.4) ensures fair exchange of feedback and services. The reputation system, oriented for content distribution peer-to-peer systems, uses cryptographic techniques to guarantee that when a peer receives the requested data blocks, he must provide positive feedback to the sender in return. Otherwise the data blocks stay locked and their content remains inaccessible to the peer.

Whitewashing

A whitewashing attack occurs when a user with negative reputation quits the system and re-enters with a new identity and thus a fresh reputation.

A reputation system is vulnerable to the whitewashing attack when: the pseudonyms in the system are not linked to real world identities, quitting the system incurs little or no loss, and creating new pseudonyms is cheap (in terms of limited resources, such as money, human effort, etc.).

To mitigate the risk of whitewashing attacks, a reputation system may differentiate users who are newcomers from those who have been in the system for a long time. A user may only be allowed to build its reputation gradually by demonstrating good behavior consistently over a long period of time. This approach lessens the appeal of a whitewashing attack, since a user who re-enters the system with a new identity is not viewed as trustworthy. Systems that propose this approach include [65, 91].

Oscillation

In oscillation, an attacker initially builds good reputation in the system and then suddenly shifts behavior to take advantage of honest users who are misled into trusting the attacker due its good reputation. This attack is advantageous only if the payoff of the attack is greater than the cost of building good reputation. One scenario is that an attacker engages in several low value transactions to accumulate reputation and then reverses its good behavior for a high value transaction. A reputation system may mitigate the risk of oscillation attacks by employing the following strategies:

- Weighing feedback according to the value of the transaction.
- Factoring the time spent in the system towards computing reputation.
- Requiring the expense of limited resources towards building reputation.
- Weighing feedback according to its age in the system. Systems that follow this strategy include Aringhieri et al. [10], Buchegger et al. [21], TrustGuard [125], and the Beta reputation system [76].

Swaminathan et al. [129] address this problem by setting a sales limit on each seller, which is bounded by the sum of the transaction costs (fees, insurance, shipping, etc.) paid by the seller thus far in the system. The seller may only sell items within its sales limit. The idea is that even if the seller suddenly shifts

behavior and defrauds a buyer, he would not make any profit due to his past expenses.

According to economic theory [84], if the benefit of having good reputation is commensurate to the cost of building the reputation, then the market reaches a state of equilibrium. In such a system there is no advantage in mounting an oscillation attack, since the attacker would loose as much as he had spent.

2.3 Privacy Preserving Reputation Systems

In this thesis, we focus on how to compute reputation in a privacy preserving manner in decentralized environments. Preserving the privacy of feedback providers is straightforward in the presence of a trusted central authority: Each provider submits his feedback value to the central authority who aggregates all feedback and reveals the reputation score while keeping the individual feedback private. However, preserving privacy in decentralized reputation systems is not trivial, since no such universally trusted central authority is present.

In Sections 2.3.1 through 2.3.8, we discuss several systems in the literature that relate to privacy preserving reputation systems. We summarize the salient features of each work, as well as present our critique and analysis. In Section 2.3.9 we compare the systems and discuss the need for novel solutions.

2.3.1 Clifton et al. [31] – Secure Sum

Secure multi-party computation is the study of protocols that take inputs from distributed entities and aggregate them to produce outputs, while preserving the privacy of the inputs.

One of the well-known secure multi-party computation protocols is secure sum [31], which takes inputs from entities and computes their sum. This protocol is clearly a natural fit for the problem at hand. The protocol may be used directly to compute reputation in the form of sum or mean.

Secure Sum

We describe the secure sum protocol below. *Note:* Equations 2.6 and 2.7 use modular arithmetic. All operations in these two equations are *mod* m.

The protocol assumes that there are three or more sites and there is no collusion between them. It is also assumed that the value to be computed, $v = \sum_{l=1}^{s} v_l$ lies in the range $[0..m]$. The sites are numbered as $1 \ldots s$. Site 1 generates a random number R uniformly chosen from $[0..m]$. It then sends $R + v_1 \ mod \ m$ to site 2, where v_1 is site 1's local input. Site 2 does not learn any information about v_1 since $R + v_1 \ mod \ m$ is distributed uniformly across the range $[0..m]$ due to R. For sites $l = 2 \ldots s - 1$, the protocol proceeds as follows: Site l receives:

$$V = R + \sum_{j=1}^{l-1} v_j \ mod \ m \tag{2.6}$$

Site l learns nothing since the value is distributed uniformly across $[0..m]$. Site l computes:

$$R + \sum_{j=1}^{l} v_j \ mod \ m = (v_l + V) \ mod \ m \tag{2.7}$$

Site l then sends this value to site $l+1$. Eventually, site s also performs the above step. Site s sends the result back to site 1, who subtracts R from it to obtain the sum. Site 1 does not learn any of the private values due to the uniform distribution of the received result over the range $[0..m]$.

The protocol may be used to compute reputation as the sum of the feedback values provided as private inputs by the participants of the protocol.

Critique: The security of the secure sum protocol breaks down if the sites collude. Any two sites $l-1$ and $l+1$ can use the values that they send and receive respectively to compute the private input v_l of site l.

Table 2.1: Clifton et al. [31] – Secure Sum.

Architecture	Decentralized
Target Environment	Distributed environments
Adversarial Model	Semi-Honest + Agents do not collude
Key Security Mechanisms	Secure multi-party computation
Privacy Guarantee	The chances that the adversary will learn private information are no better than making a random guess over the range $[0..m]$. Probability: $\frac{1}{m+1}$
Complexity (Messages)	$O(n)$, where n = number of sites

Other Secure Multi-Party Computation Protocols

Other secure multi-party computation protocols include: secure product [31, 8, 11, 74], secure set union [31, 83], secure set intersection [31, 83], and secure multiset operations [83]. The doctoral thesis of Wenliang Du [43] describes several secure two-party computation protocols for problems in linear programming, geometry, and statistical analysis. A seminal work in secure multi-party computation is the study of the Millionaire's problem [140], in which two parties must determine whose number is larger without disclosing their numbers. We refer the reader to [56] for a comprehensive study of secure multi-party computation.

2.3.2 Pavlov et al. [105] – Decentralized Additive Reputation Systems

Pavlov et al. [105] propose several protocols for decentralized additive reputation systems. Two of their protocols are secure under the semi-honest and the malicious adversarial models respectively. The protocols draw their strength from witness (feedback provider) selection schemes, which guarantee the inclusion of a certain number of honest witnesses as participants. The security mechanisms used in the protocols include secure multi-party computation, secret sharing, and discrete log commitment.

Problem Setting

A querying agent consults a group of n witnesses to compute the reputation of a target agent, where $0 < n < N$, and $N > 1$ is the number of potential witnesses. $b < N$ is the number of dishonest agents in N.

Decentralized Additive Reputation Systems

A decentralized additive reputation system is described in the article as a reputation system that satisfies the following two requirements: 1) feedback collection, combination, and propagation are implemented in a decentralized way; 2) combination of feedbacks provided by agents is calculated in an additive manner. The Beta reputation system [76] is cited as an example. The eBay reputation system is additive, however, not decentralized.

Impossibility of Perfect Privacy

The paper argues that it is impossible to guarantee perfect privacy for an honest feedback provider in a decentralized additive reputation protocol. The argument is that a dishonest agent may deterministically create a set of n feedback providers, with $n - 1$ dishonest agents and the one honest agent under attack. Given the inputs of the $n - 1$ dishonest agents and the output (the reputation score), the secret feedback of the honest agent is easily obtained.

Critique: The impossibility argument does not apply to protocols in which an honest agent may choose not to contribute his feedback. The argument also does not apply to protocols in which the set of feedback providers cannot be created deterministically.

Witness Selection Scheme 1 (WSS-1)

A witness selection scheme for a reputation protocol is a process that results in the creation of a set of witnesses. The witnesses in the set contribute their feedback towards computing the reputation of the target agent.

The first scheme [105, Lemma 2] guarantees that if honest agents are uniformly distributed over N, then at least two honest witnesses will be selected

with probability greater than $(1 - \frac{1}{n})(\frac{N-b-1}{N-1})$. The scheme is secure under the semi-honest adversarial model, in which all agents follow the protocol correctly.

According to our analysis, the complexity of the number of messages exchanged is linear in terms of the number of potential witnesses: $O(N)$. After each witness is selected, it is probabilistically decided whether to add more witnesses, therefore the count may run up to N. If each agent sends its successor the current set of witnesses, the total bandwidth utilized is $O(N^2)$.

Critique: The complexity of the scheme is a function of the population size of the potential witnesses (N) instead of the witnesses who contribute their feedback (n). The scheme also has the potential of leaving out many honest witnesses from the reputation protocol. Moreover, the scheme works only if $b < n - 1$, because otherwise $n - 1$ dishonest witnesses can select themselves into the set if the first witness selected is dishonest. Even then the scheme might fail since the number of witnesses selected is probabilistic and it may be the case that the actual number of selected witnesses is less than n.

Witness Selection Scheme 2 (WSS-2)

The second scheme [105, Lemma 3] guarantees under the malicious adversarial model that if honest agents are uniformly distributed over N, then at least $n(\frac{N-b-n}{N})$ honest witnesses would be selected. A coin flipping scheme is utilized to grow the set of witnesses by selecting the next witness randomly from the available pool of witnesses. According to the paper, the scheme requires $O(n^3)$ messages among the n selected witnesses.

Critique: It is not clear if the scheme would work in case the querying agent is dishonest. If the querying agent is dishonest, it does not need to follow the protocol correctly. It can select a dishonest witness and then collectively cheat to continue selecting dishonest witnesses. After an honest victim is selected, the rest of the witnesses must be selected randomly. However, at that point the coalition of dishonest agents has already biased the set in their favor.

A Reputation Protocol based on WSS-1

In this reputation protocol, the set of source agents is created using the first witness selection scheme, which guarantees that at least two source agents are honest. Agent q chooses a random number as its secret. Each agent splits its secret into $n + 1$ shares such that they all add up to the secret. Each agent keeps the $n + 1^{th}$ share and sends its other n shares to the other n agents in the protocol such that each agent receives a unique share. Each agent then adds all shares received along with his $n + 1^{th}$ share and sends it to the querying agent. The querying agent adds all sums received and subtracts the random number to obtain the reputation score.

The protocol guarantees the privacy of an honest source agent under the semi-honest model as long as all the other $n - 1$ source agents do not collude. The probability that all other source agents will not collude is greater than $(1 - \frac{1}{n})(\frac{N-b-1}{N-1})$. The number of messages exchanged is analyzed as $O(n^2)$. We

estimate that the size of the messages exchanged is as follows: $O(n^2)$ IDs and $O(n^2)$ numbers.

Critique: The complexity is claimed to be $O(n^2)$, however, we believe it to be $O(N) + O(n^2)$ due to the utilization of the witness selection scheme.

Table 2.2: Pavlov et al. [105] – A Reputation Protocol based on WSS-1.

Architecture	Decentralized
Target Environment	Distributed environments
Adversarial Model	Semi-honest
Key Security Mechanisms	Secure multi-party computation, secret sharing
Privacy Guarantee	$(1 - \frac{1}{n})(\frac{N-b-1}{N-1})$
Complexity (Messages)	$O(N) + O(n^2)$, where N = number of potential witnesses, and n = number of selected witnesses

A Reputation Protocol based on WSS-2

This protocol uses the Pedersen verifiable secret sharing scheme [107] and a discrete log commitment method. The Pedersen scheme is resilient up to $n/2$ malicious agents. The set of source agents is created using the second witness selection scheme. It guarantees the presence of less than $n/2$ malicious agents, if $b < \frac{N}{2} - n$.

The protocol is secure under the malicious model as long as $b < \frac{N}{2} - n$. The number of messages exchanged is $O(n^3)$, due to the second witness selection scheme.

Table 2.3: Pavlov et al. [105] – A Reputation Protocol based on WSS-2.

Architecture	Decentralized
Target Environment	Distributed environments
Adversarial Model	Malicious (Disruptive)
Key Security Mechanisms	Verifiable secret sharing, discrete log commitment
Privacy Guarantee	If $b < \frac{N}{2} - n$, then the adversary does not learn any more information about the private feedback of an honest witness
Complexity (Messages)	$O(n^3)$, where n = number of witnesses

2.3.3 Gudes et al. [61] – The Knots Reputation System

Gudes et al. [61] present several schemes that augment their Knots reputation system [51] with privacy preserving features. A defining characteristic of the Knots reputation model is the notion of subjective reputation. The reputation

of a target member is computed by each querying member using a different set of feedback, thus the reputation is subjective for each querying member. The feedback that a querying member uses for computing reputation comes exclusively from the members in which he has a certain amount of pre-existing trust. An advantage of this approach is that the querying member has confidence in each of the feedback values that are used for computing reputation.

Critique: The disadvantage is that the opinion of the members whom the querying agent does not know in not taken into account. The notion of subjective reputation tends to be non-conformant with the idea of reputation, which is generally considered to be the aggregate of feedback of the community at large. The concept of subjective reputation seems closer to trust propagation than reputation.

The Knots Model

The Knots model differentiates between two types of users in the system. The *experts* in the system are the users who provide services and the *members* are users who consume those services. The reputation system is concerned with computing the reputation of the experts through the feedback provided by the members. Members have trust relationships among themselves in the context of providing reliable feedback about the experts.

$TrustSet_x(A)$ is defined as the set of members whom member A trusts to provide feedback about expert x. $TM(A, B)$ represents the amount of direct trust that a member A has in another member B. $DTE(A, x)$ is defined as the amount of direct trust that a member A has in an expert x. The subjective reputation of an expert x by a member A is computed as follows:

$$TE(A, x) = \frac{\Sigma_{\forall B \in TrustSet_x(A)} DTE(B, x) \cdot TM(A, B)}{\Sigma_{\forall B \in TrustSet_x(A)} TM(A, B)} \tag{2.8}$$

In the privacy preserving version of the Knots model, the challenge is to compute $TE(A, x)$, such that the privacy of each $DTE(B, x)$ is maintained, where $B \in TrustSet_x(A)$. The three decentralized privacy preserving schemes presented in the paper compute $\rho(A, x)$ (the numerator of the fraction in equation 2.8), such that A cannot learn any of the $DTE(B, x)$ values.

Critique: The privacy goal does not include preserving the privacy of the trust between the members (the TM values). It is limited to preserving the privacy of the feedback about the experts (the DTE values).

Reputation Scheme 1

Each member $B \in TrustSet_x(A)$ receives $TM(A, B)$ from A and then computes $E_A(DTE(B, x) \cdot TM(A, B))$ and sends it to a Trusted Third Party (TTP), Z (where $E_A(.)$ is an encryption with the public key of member A). The TTP Z relays each message to A without revealing the source member. A decrypts the messages and obtains $\rho(A, x)$.

Since A does not know the source of a message, it cannot reverse a received value to reveal the private feedback. The messages are encrypted, therefore the TTP does not learn any information either. The scheme requires $O(n)$ messages to be exchanged, where n is the cardinality of $TrustSet_x(A)$.

Critique: The scheme requires disclosure of the trust that A has in each member B. Moreover, there is heavy reliance on the TTP. If the TTP and A collude, then they can easily determine each $TM(B, x)$.

Table 2.4: Gudes et al. [61] – Reputation Scheme 1.

Architecture	Decentralized
Target Environment	Distributed environments
Adversarial Model	Semi-honest
Key Security Mechanisms	TTP, Public-key cryptography
Privacy Guarantee	If the TTP is honest, the chances that A will learn $DTE(B, x)$ are no higher than making a random guess across $\|TrustSet_x(A)\|$ given values. $B \in TrustSet_x(A)$.
Complexity (Messages)	$O(n)$, where $n = \|TrustSet_x(A)\|$

Reputation Scheme 2

Each member $B \in TrustSet_x(A)$ generates $E_A(DTE(B, x))$ and sends it to a TTP, Z. The TTP sends a randomly permuted vector of the messages to A, who decrypts the messages and obtains a vector (vector 1) of the DTE values. A then sends a vector of all values $TM(A, B)$ to Z, where $B \in TrustSet_x(A)$. Z permutes the vector (vector 2) according to the DTE vector (with respect to the order of the members). A and Z compute the scalar product of vectors 1 and 2 using a secure product protocol (such as [8]) to obtain $\rho(A, x)$.

Due to the random permutation generated by the TTP, A is unable to correlate the DTE values with individual members. The TTP does not learn any of the DTE values due to encryption. A key advantage of the scheme is that any member B does not learn $TM(A, B)$.

We analyze that the number of messages exchanged is $O(n)$, whereas the bandwidth utilized is $O(n^2)$ in terms of k-bit numbers transfered, where k is the security parameter (key length).

Critique: The privacy of the $TM(A, B)$ values is still not fully preserved since they must be disclosed to the TTP.

Reputation Scheme 3

A executes the reputation protocol for the semi-honest model from Pavlov et al. [105] to obtain $\Sigma_{\forall B \in TrustSet_x(A)} DTE(B, x)$. A sends $TM'(A, B) = TM(A, B) + Q$ to each $B \in TrustSet_x(A)$, where Q is a random number. A

Table 2.5: Gudes et al. [61] – Reputation Scheme 2.

Architecture	Decentralized
Target Environment	Distributed environments
Adversarial Model	Semi-honest
Key Security Mechanisms	TTP, Public-key cryptography, Secure product
Privacy Guarantee	If the TTP is honest, the chances that A will learn $DTE(B,x)$ are no higher than making a random guess across $\|TrustSet_x(A)\|$ given values. Moreover, B does not learn $TM(A,B)$. $B \in TrustSet_x(A)$.
Complexity (Messages)	$O(n)$, where $n = \|TrustSet_x(A)\|$

executes the secure sum protocol [31] to obtain $\Sigma_{\forall B \in TrustSet_x(A)}(TM'(A,B) \cdot DTE(B,x))$. A calculates:

$$\rho(A,x) = \Sigma_{\forall B \in TrustSet_x(A)}(TM'(A,B) \cdot DTE(B,x)) \\ -(Q \cdot \Sigma_{\forall B \in TrustSet_x(A)} DTE(B,x)) \tag{2.9}$$

This scheme has the advantage that the privacy of both the $DTE(B,x)$ values and the $TM(A,B)$ values is preserved without the presence of any TTPs. The protocol requires $O(n^2)$ messages due to the inclusion of the protocol from [105].

Table 2.6: Gudes et al. [61] – Reputation Scheme 3.

Architecture	Decentralized
Target Environment	Distributed environments
Adversarial Model	Semi-honest + Agents do not collude
Key Security Mechanisms	Secure multi-party computation
Privacy Guarantee	A does not learn more information about $DTE(B,x)$, where $B \in TrustSet_x(A)$. The chances of B learning $TM(A,B)$ are no better than its chances of guessing the random number Q from $TM'(A,B)$.
Complexity (Messages)	$O(n^2)$, where $n = \|TrustSet_x(A)\|$

Proposals for the Malicious Adversarial Model

The work also includes some proposals for augmenting the schemes for the malicious adversarial model.

Critique: The proposals are largely based on the assumption that a member who provides feedback (member B) would lack the motivation to act maliciously

if it does not know the identity of the querying member (member A). However, this assumption does not take into account the case when an attacker may want to attack the system simply to disrupt it, for example, in a denial-of-service attack.

2.3.4 Belenkiy et al. [14] – A P2P System with Accountability and Privacy

Selfish participants are a major threat to the functionality and the scalability of peer-to-peer systems. Belenkiy et al. [14] propose a content distribution peer-to-peer system that provides accountability, which makes it resilient against selfish participants. The solution is based on e-cash technology. Despite making peers accountable, the system does not compromise the privacy of the peers. The system ensures that transactions between peers remain private. The only exception is the case when there is a dispute between transacting peers.

Although the system is not directly related to reputation, we study it here because it provides insight into designing a privacy preserving P2P system using the e-cash technology. In Section 2.3.5, we will discuss a privacy preserving reputation system based on e-cash by Androulaki et al. [9].

E-Cash and Endorsed E-Cash

E-cash [28, 29] is a digital currency that offers the following properties:

Anonymity. It is impossible to trace an e-coin (the monetary unit of e-cash) to the user who spent it. This property holds even when the bank (a central entity who issues the e-coins) is the attacker.

Unforgeability. The only exception to the anonymity property is that e-cash does not guarantee the anonymity of a user who tries to double-spend an e-coin. In this case, the bank can learn the identity of the dishonest user. A forged e-coin allows the bank to trace down the user who forged it.

Fungibility. A user can use the e-coins received for services provided as payment for services received from any other user in the system.

Endorsed e-cash [23] adds the following property to e-cash:

Fair Exchange. Fair exchange means that a buyer gets the item only if the seller gets paid and vice versa.

A Currency based Model

The authors describe a peer-to-peer content distribution system inspired by BitTorrent [109]. However, the proposed system provides stronger accounting in its protocols that allow nodes to *buy* and *barter* data blocks from their neighbors in a fair manner.

The system requires the participation of two trusted entities: 1) A *bank*, which maintains an endorsed e-cash account for each user. Users are able to make deposits and withdrawals of e-coins. 2) An *arbiter*, which protects the fair exchange of e-cash for data.

A user has two options for acquiring the data blocks that it needs: 1) it can pay e-coins to users who own those data blocks; 2) or it can barter its own data blocks for the ones that it needs. To earn e-cash, a user has to offer data blocks that other users want and exchange them for e-coins. A user is prevented from being selfish since it cannot consume the service provided by the peer-to-peer system unless he contributes as well.

An unendorsed e-coin cannot be deposited into the seller's bank account until the buyer endorses it. Each unendorsed e-coin has a contract associated with it. The fair exchange takes place according to the contract. If the seller fulfills its commitments, then the unendorsed e-coin must be endorsed by the buyer or otherwise by the arbiter.

The Buy and Barter Protocols

The *buy* protocol operates as follows: Alice requests a data block from Bob. Bob encrypts the block with a random key and sends the ciphertext to Alice. Alice sends an unendorsed e-coin and a contract for the data block. If the unendorsed e-coin and the contract are formed correctly, Bob sends the decryption key for the data block to Alice. If the key decrypts the data block correctly, Alice endorses the sent e-coin, which Bob can then deposit into his account.

The protocol ensures that fair exchange of e-coins and data takes place. If Bob is dishonest and the key is incorrect, Alice does not endorse the e-coin. In case Alice is dishonest and she does not endorse the coin after receiving the key, Bob can present the arbiter with proof of his correct service (in the form of the contract and other credentials received from Alice) and have the arbiter endorse the e-coin for him.

Moreover, the privacy of the transaction is preserved since no third party involvement is required, unless there is a need for arbitration. The e-coin spent by Alice is unlinkable to her due to the anonymity provided by e-cash.

The barter protocol also provides fair exchange and privacy. Alice and Bob initially send each other an unendorsed e-coin as collateral and a contract which lets them have the arbiter endorse the coin in case the key for a bartered data block is incorrect. Alice and Bob then continue to exchange data blocks until the occurrence of fair termination or arbitration.

Endorsed e-cash requires that each received e-coin must be deposited back to the bank before it can be spent. The buy protocol therefore incurs significant overhead due to this requirement. However, the barter protocol is scalable since it does not require any involvement from the bank under normal circumstances.

Critique: The bank and the arbiter are centralized entities. This implies that the system is not fully decentralized. The two centralized entities present scalability issues (at least for the buy protocol) as well as single points of failure.

2.3.5 Androulaki et al. [9] – A Reputation System for Anonymous Networks

Androulaki et al. [9] propose a reputation scheme for pseudonymous peer-to-peer systems in anonymous networks. Users in such systems interact only through disposable pseudonyms such that their true identity is not revealed. Reputation systems are particularly important for such environments since otherwise there is little incentive for good conduct. However, reputation systems are hard to implement for these environments. One of the reasons is that a user must keep his reputation even if he cycles through many pseudonyms. Moreover, the pseudonyms must be unlinkable to the user as well as to each other even though they share the same reputation score. Another issue that arises in reputation systems for anonymous networks is that a user may lend his good reputation to less reputable users through anonymous pseudonyms.

The proposed system employs the following cryptographic building blocks: anonymous credential systems, e-cash, and blind signatures. Reputation is exchanged in the form of e-coins called *repcoins*. The higher the amount of repcoins received from other users, the higher is the reputation of the user.

Critique: The system requires the presence of a *bank*, which is a centralized entity. Additionally, the system also requires that all communication take place over an anonymous network, such as Mixnet [27] or a network using Onion routing [40]. This requirement makes the solution inaccessible to applications in non-anonymous networks.

The security goals of reputation systems for anonymous networks are different than those of privacy preserving reputation systems. The reputation systems for anonymous networks aim to hide the identity of a user who interacts and assigns feedback to others. Whereas, in privacy preserving reputation systems, the goal is to hide the feedback value assigned but not the identity of the user who assigned it. The choice between the two kinds of reputation systems depends on the security objectives of the application.

Security Model

Some of the security requirements of the reputation system are as follows:

Unlinkability. An adversary, controlling the bank and a number of corrupted users, is unable to link a pseudonym with the identity of its non-corrupted user any better than by making a random guess. Moreover, the adversary has no advantage in telling whether two pseudonyms belong to the same non-corrupted user or not.

No Over-Awarding. A user who tries to double-award (forge) a repcoin, using one or even two different pseudonyms, gets detected and his identity is revealed.

Exculpability. Any coalition of corrupted users (including the bank) is unable to falsely accuse a user of forgery in order to expose to his identity.

Reputation Unforgeability, Non-Transferability. A user cannot forge better reputation. In particular, a user U_1 cannot borrow reputation from another user U_2, unless U_2 reveals his master secret key to U_1.

Cryptographic Building Blocks

The following cryptographic building blocks are used for the construction of the scheme:

Anonymous Credential Systems. In anonymous credential systems (for example, [15, 22]), organizations grant credentials to pseudonymous identities of users. Verifiers are able to verify the authenticity of credentials in possession of users. However, neither an organization or a verifier is able to link a credential to the true identity of a user.

E-Cash. E-cash [28, 29] is a digital currency that offers the following properties: anonymity, unforgeability (or identification of double-spenders), and fungibility. Please see Section 2.3.4 for further detail. A centralized *bank* is a key player in an e-cash system.

Blind Signatures. In a blind signature scheme (for example, [28]), an entity signs a message for a user, however the entity does not learn the content of the message.

A Reputation System for Anonymous Networks

The system assumes the presence of a central entity called the bank, which is needed for implementing the above listed cryptographic schemes. The system also requires that all communication takes place over an anonymous network, for example, a Mixnet, or a network using Onion routing. The users interact with each other in a peer-to-peer manner. However, the users must also communicate with the central bank to withdraw and deposit repcoins.

From the above listed building blocks, Androulaki et al. build a reputation system in which each user has a reputation that he cannot lie about or shed. However, a user may generate as many one time pseudonyms as he needs for his transactions. All pseudonyms of a user share the same reputation. The system is robust against self-promotion attacks. Reputation is updated and demonstrated in a way such that anonymity is not compromised. The system maintains unlinkability between the identity of a user and his pseudonyms, and unlinkability among pseudonyms of the same user.

The system by Androulaki et al. follows upon the work by Dingledine et al. [39, 38, 37] on reputations systems and anonymous networks.

2.3.6 Nin et al. [101] – A Reputation System for Private Collaborative Networks

Nin et al. [101] present a reputation system that computes the reputation of a user based on the access control decisions that he makes. If a user makes good

Table 2.7: Androulaki et al. [9] – A Reputation System for Anonymous Networks.

Architecture	Decentralized
Target Environment	Peer-to-peer systems
Adversarial Model	Malicious (Disruptive)
Key Security Mechanisms	Anonymous credential systems, E-cash (bank), Blind signatures, Mixnets / Onion Routing
Privacy Guarantee	Satisfies unlinkability, no over-awarding, exculpability, and reputation unforgeability if the underlying primitives (anonymous credential system, e-cash system, and blind signatures) are secure
Complexity (Messages)	$O(1)$

access control decisions, such as granting access to legitimate users and denying access to unauthorized users, then he receives good reputation. In contrast, making dishonest access control decisions leads to bad reputation. The privacy objective of the reputation system is to keep the trust relationships between the users private.

The system operates as follows: A node keeps record of its access control decisions. Other nodes can view anonymized details of those decisions and verify if the decisions were made according to the access control rules or not. The anonymization is derived through the multiplicative homomorphic property of the ElGamal encryption scheme. Private details are not revealed to a third-party due to the anonymization.

Private Collaborative Networks

A private collaborative network is described as a network of users that has the following properties: 1) the users are connected with each other through trust relationships; 2) users own resources that can be accessed by other users if sufficient trust exists; and 3) trust relationships among users remain private.

A private collaborative network is modeled as a directed labeled graph. Edges represent trust relationships between nodes (users). Each edge is labeled with the type of trust relationship as well as the weight of the trust.

Access to each resource in the network is governed by a set of access conditions. An access condition is of the form $ac = (v, rt, d_{max}, t_{min})$, where v is the owner with whom the requester of the resource must have a direct or transitive trust relationship of type rt to gain access. d_{max} and t_{min} are the required maximum depth and minimum trust respectively to obtain access.

Each trust relationship also exists in the form of a certificate signed by the truster and the trustee. Since relationships must be kept private, a certificate itself is considered a private resource. To gain access to a resource, a requester must demonstrate to the owner, the existence of a "certificate path" linking the requester to the owner.

The Reputation Model

The reputation system assigns good reputation to a user who performs decisions in accordance with the specified access conditions. In contrast, a user who does not correctly enforce access control rules, receives lower reputation. Reputation lies in the interval $[0, 1]$.

A user can act dishonestly in two ways: 1) deny access to a resource to a legitimate requester, or 2) allow access to a resource to an unauthorized requester. The access control decision is considered wrong if it violates either of the rt, d_{max}, t_{min} parameters in the access condition. For a wrong decision that violates the trust requirement (t_{min}), the absolute difference between the minimum amount of trust required (t_{min}) and the trust computed over the certificate path is given as wd. The values arising from all such wrong decisions are given as the set $\{wd_1, \ldots, wd_{|WD_{t_A}|}\}$, where $|WD_{t_A}|$ is the number of wrong decisions.

The values in the set $\{wd_1, \ldots, wd_{|WD_{t_A}|}\}$, which represent the wrong decisions made by user A in terms of trust, are aggregated as:

$$AGt_{AC_{SET_A}} = OWA_Q(wd_1, \ldots, wd_{|WD_{t_A}|}) \tag{2.10}$$

where $AGt_{AC_{SET_A}}$ is the aggregated value of the wrong decisions with respect to trust. OWA is an Ordered Weighted Averaging function and Q is a non-decreasing fuzzy quantifier. According to the authors: "The interest of the OWA operators is that they permit the user to aggregate the values giving importance to large (or small) values".

The wrong decisions of the user that violate the depth and path requirements are aggregated as $AGd_{AC_{SET_A}}$ and $AGp_{AC_{SET_A}}$ respectively. The reputation of user A is then computed as:

$$R_A = 1 - \frac{1}{3}(AGt_{AC_{SET_A}} + AGd_{AC_{SET_A}} + AGp_{AC_{SET_A}}) \tag{2.11}$$

which implies that the mean of the aggregates of the three types of wrong decisions is subtracted from the perfect reputation of 1 to arrive at the reduced reputation of the user. The more dishonest decisions a user makes, the lower his reputation.

Anonymized Audit Files

After a user makes an access control decision, an entry about that decision is added into the user's anonymized audit file. The entry includes information such as the identity of the requester of the resource, the certificate path demonstrated by the requester, etc. However, all private information in the entry is encrypted using the ElGamal encryption scheme [46]. Therefore, a third-party who analyzes the entry is unable to acquire any information about these private elements. Due to the multiplicative homomorphic nature of the ElGamal encryption scheme, the encrypted information can be manipulated to compute

reputation. A network participant who wishes to learn the reputation of a certain user, can analyze the anonymized audit file of that user and derive the reputation score without compromising privacy.

We analyze the number of messages exchanged to compute reputation as constant ($O(1)$), since all required information is provided directly by the target node.

Critique: We believe that the following features of the reputation system are advantageous: 1) the reputation of a node is not derived from the feedback of other nodes but from objective information about its behavior (its access control decisions), and 2) a node itself manages and furnishes the evidence required for another node to judge its reputation.

However, we also observe the following issues: 1) As we understand, a node itself manages its audit file due to the absence of centralized entities and TTPs in the system. It is not clear why a dishonest node would include its bad decisions in its audit file. If the node is itself in charge of creating the file, it would only include details that lead to good reputation. 2) Reputation is computed based on the access control decisions of a user, which makes the applicability of the reputation system limited. For example, the reputation system would not work in e-commerce systems, where the reputation of a seller is based on the subjective satisfaction of the buyers.

The adversarial model is not specified in the paper, however, we estimate that the scheme would be secure only upto the semi-honest model since nodes are assumed to manage their audit files honestly.

Table 2.8: Nin et al. [101] – A Reputation System for Private Collaborative Networks.

Architecture	Decentralized
Target Environment	Private collaborative networks
Adversarial Model	Semi-honest
Key Security Mechanisms	ElGamal encryption scheme
Privacy Guarantee	Trust relationships among users remain private if the underlying encryption scheme is secure
Complexity (Messages)	$O(1)$

2.3.7 Kinateder and Pearson [81] – A Privacy-Enhanced P2P Reputation System

The decentralized reputation system proposed by Kinateder and Pearson [81] requires a Trusted Platform Module (TPM) chip at each agent. The TPM enables an agent to demonstrate that it is a valid agent and a legitimate member of the reputation system without disclosing its true identity. This permits the agent to provide feedback anonymously.

Security Goals

The reputation system sets the security requirements listed below. An attacker must not be able to:

- Provide false feedback on an honest user's behalf.
- Access an honest user's private database and modify data such as feedback, reputation, etc.
- Learn the identity of a feedback provider (which implies that a user should be able to provide feedback anonymously).

Moreover, it is required that:

- The identity of a dishonest user can be revealed if there is sufficient legal justification.

Trusted Platform

The reputation system presented in the paper relies on the Trusted Platform (TP) [99, 106] technology for security. A trusted platform is described as a secure computing platform that preserves the privacy of the user by providing the following three functionalities:

Protected Storage. Data on the TP is protected from unauthorized access.

Integrity. The TP can prove that it is running only the authorized software and no malicious code.

Anonymity. The TP can demonstrate that it is a genuine TP without revealing the identity of the user. The TP uses a pseudonym attested by a PKI Certification Authority (CA).

A Trusted Platform comprises of a Trusted Platform Module (TPM), which is a hardware device with cryptographic functions that enable the various security functionalities of the TP. The TPM is unforgeable and tamper-resistant.

System Model and Functionality

An agent in the system can take up one of following three roles at any given time: *recommender*, *requester*, and *accumulator*.

Recommender. A recommender agent has interacted directly with other agents and has feedback about them. He regularly announces the availability of feedback to other agents in the system. A recommendation comprises of the target agent's pseudonym, the recommender agent's pseudonym, and the feedback value. The recommendation is digitally signed by the recommender.

Accumulator. An accumulator agent stores feedback about other agents. However, his feedback is not based on direct experience with the target agent but formed through the feedback that he has received from other agents in the system.

Requester. A requester agent queries other agents for feedback and then locally aggregates the feedback to determine the reputation of the target agent. A requester agent propagates the query to its peer agents who in turn propagate to their peer agents. Each peer decides when to discontinue further propagation based on whether recommendations are available among its peers. The requester agent receives the feedback from the recommender and accumulator agents queried and then aggregates the feedback to learn the reputation of the target agent.

Critique: It is not elaborated how the feedback announcement and feedback query protocols work, for example, if an algorithm such as broadcast or gossip is used. As a consequence, the complexity of the protocols is not clear. Moreover, the mechanism for aggregating the feedback is not discussed.

How Security is Achieved

The security requirements are fulfilled as follows:

- An attacker is unable to provide false feedback on an honest user's behalf since each feedback is digitally signed by the recommender. A requester agent can also verify through the recommender's TP that it has not been compromised by the adversary.
- An attacker is unable to access an honest user's private database and modify data such as feedback, reputation, etc. This is achieved due to the protected data storage functionality of the TP. Therefore, a requester can be certain that the given feedback is not false.
- An attacker does not learn the true identity of a feedback provider since only pseudonyms are used. Thus, a user is able to provide feedback anonymously and without inhibition. The pseudonym is protected by the TP and the CA of the user. Moreover, the use of MIX cascades is suggested to prevent the attacker from correlating the pseudonym with the IP address of the user.
- In case of legal justification, the CA of a user can reveal his true identity.

Voss et al. [136] and Bo et al. [18] also present decentralized systems that are based on similar lines. They both suggest using smart cards as the trusted hardware modules. A later system by Kinateder et al. [82] avoids the hardware modules, however, it requires an anonymous routing infrastructure at the network level.

Critique: Consider the following scenario: A sale on an e-commerce system may result in the disclosure of the true identities of the seller and the buyer to each other (through mailing addresses etc.), even if they use anonymous pseudonyms. We must also consider that the privacy of the pseudonym itself may need to be protected. For example, if pseudonym A assigns pseudonym B negative feedback in retaliation, then B's reputation is adversely affected due to the lack of privacy of B's feedback. Better solutions include: preserving the privacy of the feedback, or using disposable pseudonyms, which a user may change after every transaction (such as in the solution by Androulaki et al. [9]).

Table 2.9: Kinateder and Pearson [81] – A Privacy-Enhanced P2P Reputation System.

Architecture	Decentralized
Target Environment	Peer-to-peer systems
Adversarial Model	Malicious (Disruptive)
Key Security Mechanisms	Trusted platform, MIX cascades, Digital signatures
Privacy Guarantee	Security goals are satisfied if the underlying primitives (trusted platform, MIX cascades, digital signatures) are secure
Complexity (Messages)	Not Provided

2.3.8 Steinbrecher [126] – Privacy-Respecting Reputation Systems within Centralized Internet Communities

Steinbrecher [126] argues that traditional cryptographic techniques such as encryption and digital signatures can provide only "technical" security guarantees. For example, encryption and digital signatures can guarantee the confidentiality and integrity of the *text* of a reply sent by an expert to a user on a self help forum. However, these techniques cannot guarantee the misbehavior of the users themselves. For example, the user might violate confidentiality by relaying the *content* of the text to a third party, or the expert may violate integrity by giving *false advice*. It is argued that trust can mitigate these risks and that reputation systems are a suitable technology for acquiring trust.

However, the author contests that the design of current reputation systems (such as the eBay reputation system) allow open access to the interests and behavior profiles of users. A third-party may acquire information such as the time and frequency of participation, interests in specific items, feedback provided etc. Moreover, it is easy to associate the pseudonym of a user with their real identity, for example, through a mailing address.

To counter this issue, Steinbrecher presents a privacy-respecting reputation system for centralized Internet communities. The system relies on simultaneous

use of multiple pseudonyms and changing them frequently to achieve anonymity and unlinkability.

A Generalized Model for Centralized Reputation Systems

The paper presents a generalized model for centralized reputation systems. Users use global pseudonyms tied to global reputations. The set of global pseudonyms at time t is considered as $P_t = \{p_{t,1}, \ldots, p_{t,m}\}$. The set of possible reputations that might be associated with a pseudonym is given as R. $(R, +)$ is a commutative group and $+$ an operator to combine elements from R independently of t. At time t_1, each pseudonym $p_{t_1,l}$ has the reputation $rep(t_1, p_{t_1,l}) \in R$, where $l \in 1 \ldots m$. After $p_{t_1,i}$ receives a rating r_{j,i,t_1} from $p_{t_1,j}$, the reputation of $p_{t_1,i}$ at time t_2 is computed as:

$$rep(t_2, p_{t_1,i}) = rep(t_1, p_{t_1,i}) + r_{j,i,t_1} \tag{2.12}$$

where $t_2 \geq t_1$, and $p_{t_1,i}$ does not receive any rating other than r_{j,i,t_1} between t_1 and t_2.

Using Pseudonyms for Unlinkability and Anonymity

The system proposes simultaneous use of multiple pseudonyms by a user. The idea is to have a separate pseudonym for each context (for example, the context of a seller on an auction site, the context of an expert on a self help forum, etc.). It is suggested that this design leads to unlinkability between the different roles of a user on the Internet.

The system permits users to regularly change their pseudonyms to achieve anonymity. A new and an old pseudonym are unlinkable from the perspective of third-parties, however, the provider (central server) is able to link the two pseudonyms. The unlinkability also assumes that a large number of pseudonyms have the same reputation.

To prevent the provider from linking new and old pseudonyms, the system suggests using a set of non-colluding trustworthy third parties who make incremental changes to the pseudonym of the user.

Steinbrecher's work on reputation and privacy also includes [122, 108]. These proposals are oriented for centralized environments as well.

Critique: An adversary may compromise unlinkability by monitoring all pseudonyms with the same reputation. The adversary can deduce that a new pseudonym with the same reputation as a recently deleted pseudonym belong to the same user.

2.3.9 Discussion

Tables 2.11 and 2.12 provide a comparison of the reputation systems that aim to preserve privacy under the semi-honest adversarial model and the disruptive malicious adversarial model respectively.

Table 2.10: Steinbrecher [126] – A Centralized Privacy Preserving Reputation System.

Architecture	Centralized
Target Environment	E-commerce, Self-help forums, etc.
Adversarial Model	Malicious (Disruptive)
Key Security Mechanisms	Pseudonym / Identity management
Privacy Guarantee	Unlinkability and anonymity are satisfied if the provider (central server) is honest and secure
Complexity (Messages)	$O(1)$

The Semi-Honest Adversarial Model

Table 2.11: Literature – Privacy under the Semi-Honest Adversarial Model.

System / Protocol	**Architecture**	**Target Environment**	**Key Security Mechanisms**	**Privacy Guarantee**	**Complexity (Messages)**
Clifton et al. [31] – Secure Sum	D	Distributed environments	Secure multi-party computation	Probability: $\frac{1}{m+1}$, only if nodes don't collude	$O(n)$, where n = number of sites
Pavlov et al. [105] – WSS-1	D	Distributed environments	Secure multi-party computation, secret sharing	$(1 - \frac{1}{n})(\frac{N-b-1}{N-1})$	$O(N) + O(n^2)$, where N = no. of potential witnesses, and n = no. of selected witnesses
Gudes et al. [61] – Scheme 1	D	Distributed environments	TTP, Public-key cryptography	Random guess across $\lvert TrustSet_x(A)\rvert$	$O(n)$, where $n = \lvert TrustSet_x(A)\rvert$
Gudes et al. [61] – Scheme 2	D	Distributed environments	TTP, Public-key cryptography, Secure product	Random guess across $\lvert TrustSet_x(A)\rvert$	$O(n)$, where $n = \lvert TrustSet_x(A)\rvert$
Gudes et al. [61] – Scheme 3	D	Distributed environments	Secure multi-party computation	A does not learn more information about $DTE(B, x)$, where $B \in TrustSet_x(A)$	$O(n^2)$, where $n = \lvert TrustSet_x(A)\rvert$
Nin et al. [101]	D	Private collaborative networks	ElGamal encryption scheme	If the underlying encryption scheme is secure	$O(1)$

The Secure Sum protocol is simple and efficient. However, secure sum is

secure only under a restricted semi-honest adversarial model where the entities are not allowed to collude. The protocol is therefore not suitable for preserving privacy under the more realistic model where collusion is possible.

The schemes 1 and 2 by Gudes et al. provide security under the full semi-honest model. However, both schemes rely on Trusted Third Parties (TTPs). The issue with TTPs is that if they are not fully honest, they can learn private data with little or no effort.

The reputation system by Nin et al. is very efficient. It requires exchange of a constant number of messages. However, the system is limited to Private Collaborative Networks, where reputation is computed based on the access control decisions of an entity. The reputation system is not applicable to more general areas, such as e-commerce, peer-to-peer file sharing, etc.

The protocol by Pavlov et al. (based on their first witness selection scheme) is secure under the full semi-honest model. Moreover, the protocol is general purpose, that is, it may be used for many different applications. The protocol also does not rely on any TTPs or centralized constructs. The scheme 3 by Gudes et al. has similar properties. However, both these protocols have communication complexity upwards of $O(n^2)$, which is quite expensive.

The Disruptive Malicious Adversarial Model

The reputation systems by Androulaki et al. and Steinbrecher are very efficient. They require a constant number of messages to be exchanged despite the number of feedback providers and the size of the system. However, each of these systems relies on a centralized construct. The reputation system by Androulaki et al. is based on the E-Cash system, which uses a centralized construct called the bank. Steinbrecher's reputation system has a central server as an integral part of its architecture. These centralized entities make these two systems unsuitable for fully decentralized environments.

Kinateder et al.'s reputation system provides anonymity in peer-to-peer systems under the disruptive malicious model. However, the system requires the presence of special hardware called Trusted Platform (TP) at each peer. Additionally, the system requires that messages be exchanged using MIX cascades. These requirements limit the reputation system to specialized networks where TPs are available at each peer and where MIX cascades are in use.

The protocol by Pavlov et al. (based on their second witness selection scheme) is secure under the disruptive malicious model. The protocol does not require centralized constructs or specialized networks. However, the issue with the protocol is that it needs $O(n^3)$ messages to be exchanged, which is very expensive.

Challenges

We identify the following challenges in privacy preserving reputation systems:

- To the best of our knowledge, there is no framework that provides an integrated model of trust, reputation, privacy in reputation systems, and

Table 2.12: Literature – Privacy under the Disruptive Malicious Adversarial Model.

System / Protocol	Architecture	Target Environment	Key Security Mechanisms	Privacy Guarantee	Complexity (Messages)
Pavlov et al. [105] – WSS-2	D	Distributed environments	Verifiable secret sharing, discrete log commitment	If $b < \frac{N}{2} - n$	$O(n^3)$, where n = number of witnesses
Androulaki et al. [9]	D	Peer-to-peer systems	Anonymous credential systems, E-cash (bank), Blind signatures, Mixnets / Onion Routing	If the underlying primitives (anonymous credential system, e-cash system, and blind signatures) are secure	$O(1)$
Kinateder and Pearson [81]	D	Peer-to-peer systems	Trusted platform, MIX cascades, Digital signatures	If the underlying primitives (trusted platform, MIX cascades, digital signatures) are secure	Not Provided
Steinbrecher [126]	C	E-commerce, Self-help forums, etc.	Pseudonym / Identity management	If the provider (central server) is honest and secure	$O(1)$

trust recommendation and propagation. We address this issue in the next chapter.

- The privacy preserving reputation systems discussed for the semi-honest model, either rely on TTPs or specialized networks, or require a high communication complexity of $O(n^2)$, where n is the number of feedback providers. In Chapter 4, we present several protocols for the semi-honest model, which do not depend on TTPs and specialized networks, and require a communication complexity of only $O(n)$.

- It is assumed in [105] that it is not possible to achieve perfect privacy in decentralized additive reputation systems. However, we argue (in Section 4.5.1) that perfect privacy is possible by allowing feedback providers to quantify their privacy guarantee and to abstain if it is insufficient.

- We have not found any privacy preserving reputation systems for the non-disruptive malicious model (described in Section 3.4.1). In Chapter 5, we discuss a protocol for this model.

- The privacy preserving reputation systems discussed for the disruptive malicious model, either rely on centralized constructs or specialized hardware, or require a very high communication complexity of $O(n^3)$. In Chapter 5, we present a protocol for the disruptive malicious model, which does not require centralized constructs or specialized hardware. The communication complexity of our protocol is $O(n) + O(log\ N)$, where n is the number of feedback providers and N is the number of all the entities in the system.

In Chapter 6, we will introduce and address some challenges related to trust recommendation and propagation.

Chapter 3

General Framework

We establish a general framework that provides a unified view of trust, reputation, privacy in reputation systems, and trust recommendation and propagation.

The framework is founded as a multi-agent system. The formalization of trust draws upon the definition given by sociologist Diego Gambetta, which characterizes trust as binary relational, directional, contextual, and quantifiable as subjective probability. We define the reputation of a target agent as a function that aggregates the quantification of trust of other agents in that target agent.

We adopt the Ideal-Real approach to formalize the privacy preservation property of a reputation protocol. An ideal protocol for computing reputation is one in which a Trusted Third Party (TTP) receives all inputs and then locally computes the reputation. On the other hand, a real protocol computes reputation without the participation of any TTP. The real protocol is said to preserve privacy if the adversary, with high probability, cannot obtain any more information about the private input of an agent than it can learn in the ideal protocol.

Extensions to the general framework that allow it to encompass the notions of trust recommendation and propagation are presented in Chapter 6.

3.1 Agents

We model our environment as a multi-agent environment.

The Set of all Agents

Let A denote the set of all agents in the environment. The cardinality of set A is N, that is, $|A| = N$.

3.2 Trust

Our objective in this section is to arrive at a formal definition of trust. We begin by citing the definition of trust given by sociologist Diego Gambetta [52]. We

adopt this definition as the foundation for our formal model. After identifying the various characteristics of trust with the help of Gambetta's definition, we proceed to develop our formal definition of trust. The section concludes with a graphical interpretation of the model.

3.2.1 Gambetta's Definition of Trust

As discussed in Section 2.1.1, we subscribe to the definition of trust by sociologist Diego Gambetta. The reason for this choice is the ability to quantify trust as probability, which consequently makes it possible to quantify the security guarantees of the protocols that we build. The definition is given as follows [52]:

"Trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent assesses that another agent or group of agents will perform a particular action, both before he can monitor such action (or independently of his capacity ever to be able to monitor it) and in a context in which it affects his own action."

As discussed in Section 2.1.2, we infer from Gambetta's definition that trust is binary-relational, directional, contextual, and quantifiable as subjective probability. Other characteristics of trust include: non-reflexive, asymmetric, non-transitive, and dynamic.

3.2.2 Formal Definition of Trust

We first give some notations, which are followed by the formal definition of trust in Definition 1.

The Binary Relation T

Let T denote a binary relation on the set A, such that the binary relation is non-reflexive, asymmetric, and non-transitive.

Although Gambetta's definition includes "group of agents" as an entity that may be trusted, we limit our model only to individual agents. This choice yields a simpler model, which is sufficient for discussing trust between individual agents.

The Set of all Actions

Let Ψ denote the set of all actions. Some examples of actions: "prescribe correct medicine", "repair car", "deliver product sold online", etc. We refer the reader to [70] for a comprehensive discussion of actions.

The Function *perform*

Let *perform* denote a function, such that $perform : T \times \Psi \rightarrow \{true, false\}$. Let $a, b \in A$, $(a, b) \in T$, and $\psi \in \Psi$ be an action that agent $\boldsymbol{a}$ anticipates agent $\boldsymbol{b}$ to perform. Then $perform((a, b), \psi)$ outputs $true$ if agent $\boldsymbol{b}$ performs the action ψ

anticipated by agent $\boldsymbol{a}$, or it outputs false if $\boldsymbol{b}$ does not perform the anticipated action.

In the rest of the thesis, we use the simplified notation $perform(a, b, \psi)$ instead of $perform((a, b), \psi)$.

The Subjective Probability $P(perform(a, b, \psi) = true)$

Let $a, b \in A$, $(a, b) \in T$, and $\psi \in \Psi$ be an action that agent $\boldsymbol{a}$ anticipates agent $\boldsymbol{b}$ to perform. Then let the subjective probability $P(perform(a, b, \psi) = true)$ denote agent $\boldsymbol{a}$'s belief that agent $\boldsymbol{b}$ will perform the action ψ.

The Set of all Timestamps

Let $\mathcal{T}$ denote the set of all timestamps.

The 4-ary Relation $\mathbb{U}$

Let $\mathbb{U}$ denote a 4-ary relation on the sets T, Ψ, $[0, 1]$, and $\mathcal{T}$. We call the 4-ary relation $\mathbb{U}$ as the "Trust Space".

Trust

Definition 1. ***Trust.*** *The trust of an agent* $\boldsymbol{a}$ *in an agent* $\boldsymbol{b}$ *is given as the 4-tuple* $\langle aTb, \psi, P(perform(a, b, \psi) = true), \tau \rangle \in \mathbb{U}$*, where* $a, b \in A$*,* $(a, b) \in T$*,* $\psi \in \Psi$*,* $P(perform(a, b, \psi) = true) \in [0, 1]$*, and* $\tau \in \mathcal{T}$*.*

Truster

Definition 2. ***Truster.*** *Let a 4-tuple* $\langle aTb, \psi, P(perform(a, b, \psi) = true), \tau \rangle \in \mathbb{U}$ *be the trust of an agent* $\boldsymbol{a}$ *in an agent* $\boldsymbol{b}$*, where* $a, b \in A$*,* $(a, b) \in T$*,* $\psi \in \Psi$*,* $P(perform(a, b, \psi) = true) \in [0, 1]$*, and* $\tau \in \mathcal{T}$*. Then agent* $\boldsymbol{a}$ *is said to be the truster.*

Trustee

Definition 3. ***Trustee.*** *Let a 4-tuple* $\langle aTb, \psi, P(perform(a, b, \psi) = true), \tau \rangle \in \mathbb{U}$ *be the trust of an agent* $\boldsymbol{a}$ *in an agent* $\boldsymbol{b}$*, where* $a, b \in A$*,* $(a, b) \in T$*,* $\psi \in \Psi$*,* $P(perform(a, b, \psi) = true) \in [0, 1]$*, and* $\tau \in \mathcal{T}$*. Then agent* $\boldsymbol{b}$ *is said to be the trustee.*

Context of Trust

Definition 4. ***Context of Trust.*** *Let a 4-tuple* $\langle aTb, \psi, P(perform(a, b, \psi) = true), \tau \rangle \in \mathbb{U}$ *be the trust of an agent* $\boldsymbol{a}$ *in an agent* $\boldsymbol{b}$*, where* $a, b \in A$*,* $(a, b) \in T$*,* $\psi \in \Psi$*,* $P(perform(a, b, \psi) = true) \in [0, 1]$*, and* $\tau \in \mathcal{T}$*. Then action* ψ *is said to be the context of trust.*

Quantification of Trust

Definition 5. ***Quantification of Trust.*** *Let a 4-tuple* $\langle aTb, \psi, P(perform(a, b, \psi) = true), \tau\rangle \in \mathbb{U}$ *be the trust of an agent* $\boldsymbol{a}$ *in an agent* $\boldsymbol{b}$*, where* $a, b \in A$*,* $(a, b) \in T$*,* $\psi \in \Psi$*,* $P(perform(a, b, \psi) = true) \in [0, 1]$*, and* $\tau \in \mathcal{T}$*. Then the subjective probability* $P(perform(a, b, \psi) = true)$ *is said to be the quantification of trust.*

Thus, a truster's trust in a trustee is quantified as the subjective probability that the trustee will perform the action that the truster anticipates. For example, if the truster and the trustee are an online buyer and an online seller respectively, and the context is "deliver product sold online", then the amount of trust that the buyer has in the seller would be given as the subjective probability $P(perform(buyer, seller, \text{"deliver product sold online"}) = true)$.

When the context of trust is clear, we adopt the simplified notation l_{ab} for $P(perform(a, b, \psi) = true)$, that is, $l_{ab} \equiv P(perform(a, b, \psi) = true)$.

Time of Trust

Definition 6. ***Time of Trust.*** *Let a 4-tuple* $\langle aTb, \psi, P(perform(a, b, \psi) = true), \tau\rangle \in \mathbb{U}$ *be the trust of an agent* $\boldsymbol{a}$ *in an agent* $\boldsymbol{b}$*, where* $a, b \in A$*,* $(a, b) \in T$*,* $\psi \in \Psi$*,* $P(perform(a, b, \psi) = true) \in [0, 1]$*, and* $\tau \in \mathcal{T}$*. Then the time given by the timestamp* τ *is said to be the time of trust.*

The time component of trust reflects its dynamicity. If, 1) it is understood that the trust of a truster $\boldsymbol{a}$ in a trustee $\boldsymbol{b}$ refers to its most recent update, and 2) we are not concerned with the actual time of update, then we may simplify the definition of trust by removing the time component and stating it as a tuple in a 3-ary relation on the sets T, Ψ, $[0, 1]$. The trust of a truster $\boldsymbol{a}$ in a trustee $\boldsymbol{b}$ is then given as the tuple $\langle aTb, \psi, P(perform(a, b, \psi) = true)\rangle$ or $\langle aTb, \psi, l_{ab}\rangle$.

Discussion

Our formal definition of trust captures each of the following eight characteristics identified in Section 2.1.2: binary-relational, directional, contextual, quantifiable as subjective probability, non-reflexive, asymmetric, non-transitive, and dynamic.

The trust of a truster $\boldsymbol{a}$ in a trustee $\boldsymbol{b}$ is given as a 4-tuple $\langle aTb, \psi, P(perform(a, b, \psi) = true), \tau\rangle$. T is a binary relation between agents $\boldsymbol{a}$ and $\boldsymbol{b}$, which is inherently directional. The binary relation is defined as non-reflexive, asymmetric, and non-transitive. The action ψ is considered as the context of the trust. $P(perform(a, b, \psi) = true)$ quantifies trust as subjective probability.

3.2.3 Graphical Interpretation of Trust

The trust of a truster $\boldsymbol{a}$ in a trustee $\boldsymbol{b}$, given as $\langle aTb, \psi, P(perform(a, b, \psi) = true)\rangle$, may be interpreted as the graph $G_\psi =$

$(\{a, b\}, \{(a, b)\}, \{P(perform(a, b, \psi) = true)\})$. The graph G has two vertices $\boldsymbol{a}$ and $\boldsymbol{b}$ representing the truster and the trustee respectively. The edge (a, b) corresponds to the binary relation aTb. The quantification of trust $P(perform(a, b, \psi) = true)$ is given as the weight of the edge. Action ψ is the context of the graph.

A graph has a single context. If there is trust between agents in multiple contexts then a separate graph is required for each context. We do not consider the time component in the graphical interpretation.

Web of Trust

Definition 7. ***Web of Trust.*** *Let a web of trust in context ψ be defined as a weighted directed graph $G_\psi = (A, T, [0, 1])$.*

The agents in the set A form the vertices of the graph. The binary relations between agents given as ordered pairs in the set T are the edges of the graph. The weights of the edges lie in the set $[0, 1]$. A web of trust has an action ψ as its context, which is drawn from the set Ψ.

3.3 Reputation

In this section we present our formal definitions of reputation, reputation protocols, and reputation systems.

3.3.1 Definition of Reputation

Source Agent

Definition 8. ***Source Agent.*** *An agent $\boldsymbol{a}$ is said to be a source agent of an agent $\boldsymbol{b}$ in the context of an action ψ if $\boldsymbol{a}$ has trust in $\boldsymbol{b}$ in the context ψ. In other words, agent $\boldsymbol{a}$ is a source agent of agent $\boldsymbol{b}$ in context ψ if $\langle aTb, \psi, P(perform(a, b, \psi) = true)\rangle \in \mathbb{U}$.*

The set of all source agents of an agent $\boldsymbol{b}$ in context ψ is given as $S_{b,\psi} = \{a \mid \langle aTb, \psi, P(perform(a, b, \psi) = true)\rangle \in \mathbb{U}\}$. When the context is clear, the notation S_b may be used instead of $S_{b,\psi}$. The quantification of a source agent $\boldsymbol{a}$'s trust in agent $\boldsymbol{b}$, that is $P(perform(a, b, \psi) = true)$, is referred to as feedback.

Reputation

Definition 9. ***Reputation.*** *Let $S_t = \{a_1 \ldots a_n\}$ be the set of source agents of an agent t in context ψ. This implies that each agent $a \in S_t$ has the trust $\langle aTt, \psi, P(perform(a, t, \psi) = true)\rangle$ in agent t. Let rep denote a function, such that $rep : [0, 1]_1 \times \ldots \times [0, 1]_n \to \mathbb{R}$. Then the reputation of agent t in context ψ is given as: $rep(P(perform(a_1, t, \psi) = true), \ldots, P(perform(a_n, t, \psi) = true))$, or in simplified notation: $rep(l_{a_1 t}, \ldots, l_{a_n t})$.*

The reputation of an agent t is denoted by $r_{t,\psi}$, or r_t when the context is clear.

We have defined the reputation of an agent as any function that aggregates the feedback of its source agents. In the following two sections we present possible realizations of this function.

The Function rep_+

Let function rep_+ denote the reputation of an agent t in context ψ, such that:

$$rep_+(l_{a_1 t} \ldots l_{a_n t}) = \sum_{i=1}^{n} l_{a_i t} \tag{3.1}$$

The function rep_+ implements the reputation of an agent t as the sum of the feedback values of its source agents.

The Function $rep_\oplus$

Let function $rep_\oplus$ denote the reputation of an agent t in context ψ, such that:

$$rep_\oplus(l_{a_1 t} \ldots l_{a_n t}) = \frac{\sum_{i=1}^{n} l_{a_i t}}{n} \tag{3.2}$$

The function $rep_\oplus$ implements the reputation of an agent t as the mean of the feedback values of its source agents. We will use this realization of reputation in the upcoming chapters. The reason for this choice is that mean is a statistic which is intuitive and easy to understand for human users. The eBay reputation system (`ebay.com`), which is one of the most successful reputation systems, represents reputation as the simple sum of all feedback. Our decision to define reputation in simple and intuitive terms is influenced substantially by the success of the eBay reputation system. However, we go one step further and derive the average from the sum in order to normalize the reputation values.

Please note that $rep_\oplus$ is essentially a function of rep_+. The protocols that we develop in subsequent chapters for computing $rep_\oplus$, first compute rep_+ and then use the result to obtain $rep_\oplus$.

3.3.2 Reputation Protocols

Reputation Protocol

Definition 10. ***Reputation Protocol.*** *Let* Π *be a multi-party protocol. Then* Π *is defined as a Reputation Protocol, if 1) the participants of the protocol include: agents* q*,* t *and all* n *source agents of* t *in the context* ψ*, 2) the inputs include: the feedbacks of the source agents in context* ψ*, and 3) the output of the protocol is: agent* q *learns the reputation* $r_{t,\psi}$ *of agent* t*.*

Querying Agent

Definition 11. ***Querying Agent.*** *When an agent q initiates a reputation protocol to determine the reputation of an agent t, we refer to agent q as the querying agent.*

Target Agent

Definition 12. ***Target Agent.*** *When an agent q initiates a reputation protocol to determine the reputation of an agent t, we refer to agent t as the target agent.*

Decentralized Reputation Protocol

Definition 13. ***Decentralized Reputation Protocol.*** *A reputation protocol Π is said to be a decentralized reputation protocol, if there does not exist an agent C (a central entity) whose participation is required in every execution of the protocol.*

3.3.3 Reputation Systems

Reputation System

Definition 14. ***Reputation System.*** *A reputation system R is defined as a pair (A, Π), where A is the set of all agents in the environment, and Π is a reputation protocol whose participants are in A.*

Decentralized Reputation System

Definition 15. ***Decentralized Reputation System.*** *A reputation system $D = (A, \Pi)$ is defined as a decentralized reputation system, if Π is a decentralized reputation protocol.*

3.4 Adversary

The goal of a multi-party protocol is to compute the specified output from the inputs of the participants. All participants of the protocol are expected to pursue this and only this goal. An honest participant is one who conforms to this expectation. However, there may exist dishonest participants who have ulterior motives. Those motives may include learning the inputs of other participants, tampering with the output, disrupting the protocol, etc.

Unless stated otherwise, we assume that the dishonest agents in a protocol collude to achieve their ulterior objectives. We refer to this coalition of dishonest agents as the adversary.

We identify three adversarial models, which characterize the behavior of dishonest agents. The models are: Semi-Honest, Non-Disruptive Malicious, and Disruptive Malicious. When developing a protocol, we will assume that all dishonest agents operate under one of these three models.

The semi-honest and the disruptive malicious models are standard adversarial models [56]. We also consider an intermediate model that we call the non-disruptive malicious model. The non-disruptive malicious model adds an extra layer of granularity between the standard semi-honest and disruptive malicious models. We describe the three adversarial models in the following subsection.

3.4.1 Adversarial Models

Semi-Honest. In the semi-honest model, the agents do not deviate from the specified protocol. In other words, they always execute the protocol according to the specifications. The adversary abstains from wiretapping and tampering of the communication channels. However, within these constraints, the adversary passively attempts to learn the inputs of honest agents by using intermediate information received during the protocol and any other information that it can gain through other legitimate means.

Non-Disruptive Malicious. Malicious agents are not bound to conform to the protocol. Agents under a malicious model may deviate from the protocol as and when they deem necessary. They actively attempt to achieve their objectives. They may participate in extra-protocol activities, devise sophisticated strategies, and exhibit arbitrary behavior. Specifically, malicious agents may 1) refuse to participate in the protocol, 2) provide out of range values as their inputs, 3) selectively drop messages that they are supposed to send, 4) prematurely abort the protocol, 5) distort information, and 6) wiretap and tamper with all communication channels.

We define a non-disruptive malicious adversary as a malicious adversary who executes the malicious actions only if they lead to the disclosure of the inputs of honest agents. Non-disruptive agents have a single objective: learn the inputs of honest agents. They do not disrupt the normal function of the protocol other than to achieve this objective.

Disruptive Malicious. We define a disruptive malicious adversary as a malicious adversary who has the following objectives: 1) learn the inputs of honest agents, and 2) disrupt the protocol for honest agents. The reasons for disrupting the protocol may range from gaining illegitimate advantage over honest agents to completely denying the service of the protocol to honest agents.

3.4.2 General Limitations of the Adversary

We assume the adversary to be limited in the aspects defined below.

Non-Adaptive. We consider that the adversary is non-adaptive. This means that the set of dishonest agents is fixed before the execution of the protocol and it remains fixed throughout the execution of the protocol. However, the honest agents do not know the members of the set of dishonest agents.

Computationally PPT Bounded. We assume that the computational power of the adversary is Probabilistic Polynomial Time (PPT) bounded. Intuitively, a computationally PPT bounded adversary can only run an algorithm that takes polynomial amount of time. The adversary is thus unable to break any cryptographic scheme that requires more than polynomial time to do so. We refer the reader to [56] for a discussion of computationally PPT bounded adversaries.

3.5 Security

We consider a protocol to be secure under a given adversarial model if it satisfies the following two requirements under that adversarial model: correctness and privacy preservation.

3.5.1 Correctness

A protocol satisfies the correctness requirement if it always produces an output consistent with its specification. This means that the value of the output is correct, and that all recipients learn the correct value.

Let us consider a protocol Π which takes input x_i from each participant a_i, where $i \in \{1 \dots n\}$. The protocol is stipulated to output the sum of all inputs, that is, $\sum x_i$, to all participants. Then the protocol Π is said to be correct if upon every execution, each participant learns the output y and $y = \sum x_i$.

3.5.2 Privacy

We first give our definitions of some important concepts regarding privacy. We then proceed to describe the privacy preservation property of a protocol.

Private Data

Definition 16. ***Private Data.*** *Let x be some data and an agent $\mathbf{a}$ be the owner of x. Then x is agent $\mathbf{a}$'s private data if agent $\mathbf{a}$ desires that no other agent learns x. An exception is those agents to whom $\mathbf{a}$ reveals x herself. However, if $\mathbf{a}$ reveals x to an agent $\mathbf{b}$, then $\mathbf{a}$ desires that $\mathbf{b}$ does not use x to infer more information. Moreover, $\mathbf{a}$ desires that $\mathbf{b}$ does not reveal x to any third party.*

Preservation of Privacy

Definition 17. ***Preservation of Privacy (by an Agent).*** *Let x be an agent $\mathbf{a}$'s private data that agent $\mathbf{a}$ reveals to an agent $\mathbf{b}$. Then agent $\mathbf{b}$ is said to preserve the privacy of agent $\mathbf{a}$ w.r.t. x, if 1) $\mathbf{b}$ does not use x to infer more information, and 2) $\mathbf{b}$ does not reveal x to any third party.*

The Action ρ

Let action ρ = "preserve privacy". The action "preserve privacy" is synonymous with the action "be honest", since an agent preserves privacy only if it is honest, and an honest agent always preserves privacy since it has no ulterior motives.

Trusted Third Party (TTP)

Definition 18. ***Trusted Third Party (TTP).*** *Let $S \subseteq A$ be a set of n agents, and $TTP_S \in A$ be an agent. Then TTP_S is a Trusted Third Party (TTP) for the set of agents S if for each $a \in S$, $P(perform(a, TTP_S, \rho) = true) = 100\%$.*

Thus, a Trusted Third Party (TTP) for a set of agents is an entity whom every agent in the set fully trusts to preserve its privacy.

We now give an intuitive description of the privacy preservation property of a protocol.

Privacy Preserving Protocol

We adopt the Ideal-Real approach [56, 25, 24] to formalize the privacy preservation property of a protocol.

Intuitively, a multi-party protocol in the Ideal Model is a protocol that comprises of a TTP as a participant. The TTP receives all inputs in the protocol and then locally computes the output. On the other hand, a multi-party protocol in the Real Model is a protocol that does not rely on a TTP, and computes the output in a distributed manner.

The privacy preservation property of a real protocol $\mathcal{R}$ is formalized as follows: An ideal protocol $\mathcal{I}$ is first defined which has the same functionality as the protocol R. This means that the ideal protocol has the same parameters (participants, inputs, outputs, etc.) as the protocol $\mathcal{R}$, with the exception of a TTP as an additional participant. In the ideal protocol, the TTP receives the private inputs of the participants, computes the output, and sends it to the recipients. It is assumed that the privacy of the participants is preserved since the TTP is honest and fully trusted. In an ideal protocol the adversary cannot obtain any more information about the private input of a participant other than what it can learn from the information that it has beforehand and the output of the protocol that it receives.

The real protocol, which cannot consider a TTP as a participant, is said to preserve privacy if it can emulate the ideal protocol. Emulating the ideal protocol means that the adversary, with high probability (see Section 3.6), cannot obtain any more information about the private input of an agent than it can learn in the ideal protocol.

3.6 Privacy Preserving Reputation Protocols

Security Threshold

The *security threshold* is a parameter that can be assigned a value in $[0, 1]$ according to the security needs of an application. A value of the *security threshold* closer to 1 indicates a stricter security requirement. For example, in the experiments (Section 4.8) on the Advogato.org web of trust, the *security threshold* is realized as 0.90. We consider *high* probability as probability greater than or equal to the *security threshold*, and *low* probability as probability less than the *security threshold*.

Ideal Privacy Preserving Reputation Protocol

Definition 19. ***Ideal Privacy Preserving Reputation Protocol.*** *Let* Π *be a reputation protocol. Then* Π *is an ideal privacy preserving reputation protocol under a given adversarial model, if: 1) the inputs of all* n *source agents of* t *are private, 2)* TTP_{S_t} *is a participant, where* $S_t = S_{t,\psi}$ *is the set of all source agents, 3)* $m < n$ *of the source agents (given as set* M*) and agents* q *and* t *are considered to be dishonest, however,* q *wishes to learn the correct output, 4) agents* $S_t - M$ *and* TTP_{S_t} *are honest, 5) as part of the protocol, the* TTP_{S_t} *receives the private inputs from the source agents and outputs the reputation* $r_{t,\psi}$ *to agent* q*, and 6) over the course of the protocol, the private input of each agent* $a \in S_t - M$ *may be revealed only to the* TTP_{S_t}*.*

In an ideal privacy preserving reputation protocol, it is assumed that for each agent $a \in S_t - M$*, the adversary does not gain any more information about the private input of agent* $\boldsymbol{a}$ *from the protocol other than what he can deduce from what he knows before the execution of the protocol and the output, with probability* $P(perform(a, TTP_{S_t}, \rho) = true) = 100\%$*, under the given adversarial model.*

Real Privacy Preserving Reputation Protocol

Definition 20. ***Real Privacy Preserving Reputation Protocol.*** *Let* $\mathcal{I}$ *be an ideal privacy preserving reputation protocol. Then* $\mathcal{R}$ *is a real privacy preserving reputation protocol w.r.t.* $\mathcal{I}$ *under a given adversarial model, if: 1)* $\mathcal{R}$ *has the same parameters (participants, private inputs, output, adversary, honest agents, setup, etc.) as* $\mathcal{I}$*, except that there is no* TTP_{S_t} *as a participant 2) the adversary learns no more information about the private input of any agent* $\boldsymbol{a}$ *than it can learn in protocol* $\mathcal{I}$*, with high probability, when both protocols are operating under the given adversarial model.*

3.7 Communication

The communication model is considered to have the characteristics listed below. These are standard assumptions for defining secure protocols [56].

Point-to-Point. A point-to-point communication channel exists between every two parties in the network.

Synchronous. The communication is synchronous.

A malicious adversary may wire-tap and tamper with the communication channels. We describe these two malicious actions as follows:

Wire-Tapping. Wire-tapping of a communication channel between two honest parties implies that the adversary is able to obtain all messages transmitted between the two parties.

Tampering. Tampering with a communication channel between two honest parties comprises of malicious activities such as modifying, duplicating, and generating messages on a communication channel between the two parties.

3.8 Discussion

Our framework has the following characteristics:

- The framework provides a unified view of trust, reputation, preserving privacy in reputation protocols, and trust recommendation and propagation.
- Trust is defined as contextual and an action is considered as the context of trust. Trust is quantified as the subjective probability that the trustee will perform the action. Moreover, trust is not considered to be necessarily reflexive and transitive.
- The reputation of an agent is defined as any function that aggregates the feedback of its source agents. Realizations of the feedback aggregation function include summation and mean.
- The framework considers three adversarial models: semi-honest, non-disruptive malicious, and disruptive malicious.
- The privacy preservation property of a reputation protocol is formalized using the Ideal-Real approach.
- The framework is general, that is, it is not coupled with any specialized hardware platforms or networks.

There are a number of frameworks in the literature that model trust (Sant and Maple [120], Capra [26], Abdul-Rahman and Hailes [1], Marsh [90]), reputation (Levien [87], Kamvar et al. [80], Josang and Ismail [76]), and privacy in reputation systems (Nin et al. [101], Androulaki et al. [9], Pavlov et al. [105], Kinateder and Pearson [81]). However, these frameworks do not provide an integrated model of trust, reputation, privacy in general decentralized reputation systems, and trust recommendation and propagation. As discussed in

Section 2.3, the frameworks provided by Nin et al. [101], Androulaki et al. [9], and Kinateder and Pearson [81] are specific to private collaborative networks, anonymous networks, and trusted platforms respectively. The framework given by Pavlov et al. [105] integrates trust, reputation, and privacy. However, their model differs from ours as they do not quantify trust as subjective probability and they do not formalize the privacy preservation property of a reputation protocol using the Ideal-Real approach.

Chapter 4

Reputation Protocols for the Semi-Honest Adversarial Model

In this chapter, we propose some reputation protocols that are decentralized and secure under the semi-honest adversarial model (discussed in Section 3.4.1).

Our first step is to define the problem. We then give an ideal privacy preserving reputation protocol, which is considered secure by definition. The privacy preservation property of the subsequent real protocols is established by comparing them with this ideal protocol.

We then proceed to develop and analyze a simple real privacy preserving reputation protocol based on the secure sum protocol [31]. Although the protocol is secure only under a restricted semi-honest model, it gives insight into the challenges faced in developing a real privacy preserving reputation protocol.

The main contributions of this chapter are two privacy preserving reputation protocols (Round-Trip and k-Shares), which provide security under the full semi-honest adversarial model.

The Round-Trip protocol (Section 4.4) builds upon the secure multi-party computation techniques of the secure-sum protocol. Additional techniques utilized by the protocol include trust awareness, data perturbation, and a "round-trip" of messages instead of messages forwarded in a single direction. In Section 4.5, we present two extensions to the round-trip protocol. The first extension allows agents to abstain from providing feedback when the privacy guarantee offered by the protocol is insufficient. The second extension introduces *seed* agents, which further strengthens the protocol.

The k-Shares protocol (Section 4.6) combines trust awareness with secret sharing. Compared to Round-Trip, the k-Shares protocol provides each agent with a greater choice of trustworthy agents to rely on. In Section 4.7, we propose algorithms that determine the suitability of the k-shares protocol for a given complete or partial web of trust.

We present the experimental results in Section 4.8 and a comparison of our protocols with the existing protocols in Section 4.9.

4.1 Problem Definition

Definition 21. ***Problem Definition (Semi-Honest).*** *Let $S_{t,\psi} = \{a_1 \dots a_n\}$ be the set of all source agents of agent t in the context of action ψ. Find a reputation protocol Π, which takes private input $l_{at} \equiv P(perform(a, t, \psi) = true)$ from each agent $a \in S_t$, and outputs the reputation $r_{t,\psi}$ of the target agent t to a querying agent q. Reputation is computed as $rep_{\oplus}$ (Equation 3.2). Agents q, t, and $m < n$ of the source agents (given as set $\mathbb{M}$) are considered to be dishonest, however, q wishes to learn the correct output. The reputation protocol Π is required to be decentralized and secure under the semi-honest model.*

4.2 An Ideal Reputation Protocol

We first present an ideal privacy preserving reputation protocol. The subsequent protocols in the real model will attempt to emulate the privacy preservation property of this protocol. The protocol is straight forward: the TTP receives the private inputs of all source agents, computes the output and sends it to the querying agent. Since the agents are semi-honest, they follow the protocol according to the specification, and the adversary does not wire-tap or tamper with the communication channels.

4.2.1 Protocol Specification

The protocol is specified in Figure 4.1.

4.2.2 Security

Correctness

Theorem 1. *In the protocol Semi-Honest-Ideal (Figure 4.1), agent q learns the correct reputation of agent t in the context ψ, under the semi-honest model.*

Proof. The semi-honest adversarial model implies that the protocol is followed according to the specification and that the messages are not tampered with.

Each agent $a \in S_t$ sends l_{at} to TTP_{S_t}. TTP_{S_t} computes $r_{t,\psi} = (\sum_{i=1}^{n} l_{a_i t})/n$, which is the correct reputation of t in the context ψ (Equation 3.2). The TTP_{S_t} sends $r_{t,\psi}$ to q. Thus agent q learns the correct reputation of agent t in the context ψ, under the semi-honest model. □

Privacy

Theorem 2. *Semi-Honest-Ideal is an ideal privacy preserving reputation protocol under the semi-honest model. Moreover, the adversary does not gain any*

Protocol: Semi-Honest-Ideal

Participants: Agents: q, t, $S_t \equiv S_{t,\psi} = \{a_1 \dots a_n\}$, TTP_{S_t}. Agents q, t, and a subset of $S_{t,\psi}$ of size $m < n$ are considered to be dishonest, however, q wishes to learn the correct output. $n \geq 3$.

Input: Each source agent $\boldsymbol{a}$ has a private input $l_{at} \equiv P(perform(a, t, \psi) = true)$.

Output: Agent q learns $r_{t,\psi}$, the reputation of agent t in the context ψ.

Setup: Each agent $\boldsymbol{a}$ maintains $S_a \equiv S_{a,\psi}$, the set of its source agents in the context ψ. All participants know the identity of TTP_{S_t}.

Steps:

1. The querying agent q sends a request for the set S_t to the target agent t.
2. Agent q receives the set S_t from agent t.
3. Agent q sends the set S_t and the identity of agent t to TTP_{S_t}.
4. Agent q sends the identity of agent t and ψ to each agent $a \in S_t$.
5. Each agent $a \in S_t$ sends l_{at} to TTP_{S_t}.
6. TTP_{S_t} after receiving l_{at} from each agent $a \in S_t$, computes the reputation of agent t as $r_{t,\psi} = (\sum_{i=1}^{n} l_{a_i t})/n$.
7. TTP_{S_t} sends $r_{t,\psi}$ to q.

Figure 4.1: Protocol: Semi-Honest-Ideal

more information about the private input of each agent $a \in S_t - \mathbb{M}$ other than what he can deduce from what he knows before the execution of the protocol and from the output, with probability $P(perform(a, TTP_{S_t}, \rho) = true)$, under the semi-honest model.

Proof. The semi-honest adversarial model implies that the protocol is followed according to the specification, and the communication channels are not wire-tapped and not tampered with.

Semi-Honest-Ideal is an ideal privacy preserving reputation protocol (Definition 19) under the semi-honest model since: 1) the inputs of the n source agents of t are private, 2) TTP_{S_t} is a participant, where S_t is the set of all source agents, 3) $m < n$ of the source agents (given as set $\mathbb{M}$) and agents q and t are considered to be dishonest, however, q wishes to learn the correct output, 4) agents $S_t - \mathbb{M}$ and TTP_{S_t} are honest, 5) as part of the protocol, the TTP_{S_t} receives the private inputs from the source agents and outputs the reputation $r_{t,\psi}$ to agent q (the source agents and TTP_{S_t} follow the protocol according to the specification), and 6) over the course of the protocol, the private input of each agent $a \in S_t - \mathbb{M}$ is revealed only to the TTP_{S_t} (the adversary cannot wire-tap the communication channels).

It follows from Definition 19, that the adversary does not gain any more information about the private input of each agent $a \in S_t - \mathbb{M}$ other than what

he can deduce from what he knows before the execution of the protocol and from the output, with probability $P(perform(a, TTP_{S_t}, \rho) = true)$, under the semi-honest model. □

4.2.3 Complexity

Table 4.1: Protocol Semi-Honest-Ideal – Complexity.

Steps	**Messages**	**IDs**	**Numbers**
1	1		
2	1	n	
3	1	$n+1$	
4	n	n	
5	n		n
6			
7	1		1
Total	$2n+4$	$3n+1$	$n+1$
Complexity	$O(n)$	$O(n)$	$O(n)$

The protocol requires $2n+4$ messages to be exchanged (complexity: $O(n)$).

In terms of bandwidth used, the protocol requires transmission of the following amount of information: $3n+1$ agent IDs (complexity: $O(n)$), and $n+1$ numbers (complexity: $O(n)$).

4.3 The Secure Sum Reputation Protocol

The secure sum protocol [31], [134] is a well known protocol that computes the sum of local values of multiple sites without revealing the local value of any site. The protocol is clearly applicable to the problem of computing the reputation of an agent in an additive manner while preserving the privacy of local feedback values. However, the secure sum protocol has the shortcoming that it does not preserve privacy when the sites are allowed to collude. We include the secure sum protocol here to establish a sense of the challenges faced in developing a privacy preserving reputation protocol for the real model. Moreover, our Round-Trip protocol (Section 4.4) is inspired by the secure sum protocol.

4.3.1 Tools and Techniques: Data Perturbation

Data perturbation is a technique for hiding a data item by adding noise to it. The noise added is sufficiently large in order to make the derivation or estimation of the data item from the resulting sum highly improbable.

We quote the Data Perturbation Assumption from [43] as follows (variable names differ from the original definition):

Definition 22. ***Data Perturbation Assumption.*** *If an input is $x \in X$, we assume that $x + y$ effectively preserves the privacy of x if y is a secret random number uniformly distributed in a domain Y, where $|Y| \gg |X|$.*

As an example, let's consider that a value $x = 0.5 \in [0, 1]$ is to be hidden. If we add a secret random number $y = 3.2 \in [0, 10]$ to x, then the sum $x + y = 3.7$. In this case it is impossible to learn any information about x from the sum.

The data perturbation technique is well established in several domains including privacy-preserving data mining [4], [134], and secure two party [43], [55] and multi-party [55] computation.

With data perturbation, there is some probability that x will not be hidden completely. In the above example, if $x = 1$ and the secret random number turns out to be $y = 10$, then the sum would be $x + y = 11$, which would give away the value of x. As another case, if $x = 1$ and the secret random number is $y = 9.5$, then the sum would be $x + y = 10.5$, which informs that $x \geq 0.5$.

For $x \in [0, 1]$ and $y \in [0, \Upsilon]$, we state the data perturbation assumption as follows:

Definition 23. ***Data Perturbation of*** $x \in [0, 1]$ ***with*** $y \in [0, \Upsilon]$***.*** *We assume that for $x \in [0, 1]$, the interval $[0, \Upsilon]$ is large enough that when a secret random number y, uniformly distributed in $[0, \Upsilon]$, is added to x, the probability that x will be completely hidden is high.*

4.3.2 Protocol Specification

A variant of the secure sum protocol adapted for computing the reputation of a target agent t is specified in Figure 4.2.

4.3.3 Security

Correctness

Theorem 3. *In the protocol Semi-Honest-Secure-Sum (Figure 4.2), agent q learns the correct reputation of agent t in the context ψ, under the semi-honest model.*

Proof. The semi-honest adversarial model implies that the protocol is followed according to the specification and that the messages are not tampered with.

Agent q sends $\overrightarrow{S_t}$ and a random number y to a_1. Agent a_1 adds $l_{a_1 t}$ to y and sends the sum σ_1 to a_2. Subsequently, each agent $a_i \in \{a_2 \ldots a_n\}$ adds $l_{a_i t}$ to the sum. Thus, when a_n sends the total σ_n to q, $\sigma_n = y + \sum_{i=1}^{n} l_{a_i t}$. q computes $r_{t,\psi} = (\sigma_n - y)/n$, which is the correct reputation of agent t (Equation 3.2). Thus the execution of the protocol results in q learning the correct reputation of agent t in the context ψ, under the semi-honest model. □

Protocol: Semi-Honest-Secure-Sum

Participants: Agents: q, t, $S_t \equiv S_{t,\psi} = \{a_1 \dots a_n\}$. Agents q, t, and a subset of $S_{t,\psi}$ of size $m < n$ are considered to be dishonest, however, q wishes to learn the correct output. $n \geq 3$.

Input: Each source agent $\boldsymbol{a}$ has a private input $l_{at} \equiv P(perform(a, t, \psi) = true)$.

Output: Agent q learns $r_{t,\psi}$, the reputation of agent t in the context ψ.

Setup: Each agent $\boldsymbol{a}$ maintains $S_a \equiv S_{a,\psi}$, the set of its source agents in the context ψ.

Steps:

1. The querying agent q sends a request for the set S_t to the target agent t.
2. Agent q receives the set S_t from agent t.
3. Agent q creates an ordered list of the agents in S_t, which is given as the vector $\overrightarrow{S_t} = \langle a_1, a_2, \dots, a_n \rangle$.
4. Agent q sends the identity of agent t, the vector $\overrightarrow{S_t}$, and a random number $y \in [0, \Upsilon]$ to agent a_1.
5. Agent a_1 computes $\sigma_1 = y + l_{a_1 t}$. Agent a_1 then sends the identity of agent t, vector $\langle a_2, a_3, \dots, a_n \rangle$, and σ_1 to a_2.
6. Each subsequent agent a_i receives the identity of agent t, vector $\langle a_i, a_{i+1}, \dots, a_n \rangle$, and σ_{i-1}. Agent a_i then computes $\sigma_i = \sigma_{i-1} + l_{a_i t}$ and sends the identity of agent t, vector $\langle a_{i+1}, a_{i+2}, \dots, a_n \rangle$, and σ_i to a_{i+1}.
7. The last agent a_n upon receipt of the identity of agent t, vector $\langle a_n \rangle$, and σ_{n-1}, computes $\sigma_n = \sigma_{n-1} + l_{a_n t}$, and sends σ_n to q.
8. q computes $r_{t,\psi} = (\sigma_n - y)/n$.

Figure 4.2: Protocol: Semi-Honest-Secure-Sum

Privacy

In the secure sum protocol, the predecessor and the successor of an agent can collude to breach the privacy of the agent. An agent a_i receives σ_{i-1} from its predecessor a_{i-1}. Agent a_i then sends $\sigma_i = \sigma_{i-1} + l_{a_i t}$ to its successor a_{i+1}. If agents a_{i-1} and a_{i+1} share the values σ_{i-1} and σ_i, they can compute $l_{a_i t} = \sigma_i - \sigma_{i-1}$. Therefore, the secure sum protocol is not a real privacy preserving protocol under the full semi-honest model. The protocol preserves privacy only under a restricted semi-honest model, in which agents are not allowed to collude.

Theorem 4. *Let's add the following restriction to the semi-honest model: dishonest agents do not collude. Then Semi-Honest-Secure-Sum is a real privacy preserving protocol under the restricted semi-honest model.*

Proof. The semi-honest adversarial model implies that the protocol is followed according to the specification, and that the communication channels are not wire-tapped and not tampered with. Additionally, it is assumed in the restricted model that agents do not collude.

When agent a_2 receives $\sigma_1 = y + l_{a_1 t}$ from agent a_1, agent a_2 learns no information about $l_{a_1 t}$, with high probability, due to the data perturbation provided by $y \in [0, \Upsilon]$ (Definition 22). Agent a_2 has no knowledge of y and it does not collude with any other agent to be able to learn y.

Subsequently, when each agent $a_i \in \{a_2 \dots a_{n-1}\}$ sends σ_i to agent a_{i+1}, agent a_{i+1} cannot learn any information about $l_{a_j t} \in \{l_{a_1 t} \dots l_{a_i t}\}$, with high probability, since each $l_{a_j t}$ is perturbed by $y + \sum_{k \in \{1 \dots i\} - \{j\}} l_{a_k t}$. y and $l_{a_k t} \in \{l_{a_1 t} \dots l_{a_i t}\}$ are not known to a_{i+1}.

Towards the end of the protocol, when agent a_n sends $\sigma_n = y + \sum_{i=1}^{n} l_{a_i t}$ to agent q, agent q can only learn $\sum_{i=1}^{n} l_{a_i t}$ by subtracting y from σ_n. $\sum_{i=1}^{n} l_{a_i t}$ is also known to q in the Semi-Honest-Ideal protocol (as the product of $r_{t,\psi}$ and n), therefore whatever q can learn at this point can also be obtained in the ideal protocol.

The protocol Semi-Honest-Secure-Sum is a real privacy preserving reputation protocol (Definition 20) under the restricted semi-honest model, since: 1) Semi-Honest-Secure-Sum has the same parameters as Semi-Honest-Ideal (except the TTP), and 2) the adversary (each of the dishonest agents) learns no more information about the private input of an agent $\boldsymbol{a}$ in Semi-Honest-Secure-Sum than it can learn in Semi-Honest-Ideal, with high probability, under the restricted semi-honest adversarial model. □

4.3.4 Complexity

Table 4.2: Protocol Semi-Honest-Secure-Sum – Complexity.

Steps	**Messages**	**IDs**	**Numbers**
1-2	2	n	
4-7	$n+1$	$\frac{n(n+1)}{2} + n$	$n+1$
8			
Total	$n+3$	$\frac{n(n+1)}{2} + 2n$	$n+1$
Complexity	$O(n)$	$O(n^2)$	$O(n)$

The protocol requires $n+3$ messages to be exchanged (complexity: $O(n)$).

In terms of bandwidth used, the protocol requires transmission of the following amount of information: $\frac{n(n+1)}{2} + 2n$ agent IDs (complexity: $O(n^2)$), and $n+1$ numbers (complexity: $O(n)$).

4.3.5 Discussion

In the secure sum protocol, the order in which agents contribute to the sum is pre-determined by the querying agent. A source agent has no choice in the selection of its predecessor and successor. This enables the predecessor and the successor of an agent to collude and breach the privacy of the agent.

4.4 The Round-Trip Reputation Protocol

In this section, we present our round-trip protocol, which is a real privacy preserving reputation protocol under the semi-honest model. The round-trip protocol is inspired by the secure sum protocol. However, the round-trip protocol provides security under the semi-honest model without the restriction that agents do not collude.

4.4.1 Tools and Techniques: Trust Awareness

We define the trust awareness of an agent $\boldsymbol{a}$ as its knowledge of the trustworthiness of fellow agents in the system in the context of preserving privacy. Having this knowledge allows the agent to differentiate between fellow agents who are more likely to preserve its privacy. This knowledge may be acquired by an agent through any of the techniques discussed in Section 2.1.3, that is, direct interaction, trust recommendation and propagation, trust negotiation, and reputation.

4.4.2 Protocol Outline

We outline below some of the key steps of the protocol.

1. **Initiation.** The protocol is initiated by a querying agent q to determine the reputation of a target agent t. Agent q retrieves S_t from t and initiates the *forwards* trip by sending $S = S_t$ and $r = 0$ to an agent randomly selected from S_t.

2. **Forwards Trip.** In the *forwards* trip, the receiving agent adds its local feedback value and a random number y to r. After removing itself from S, the agent sends the updated S and r to the agent in S that it trusts the most to preserve its privacy. The protocol continues with the *forwards* trip in this manner until the last agent in S updates r. The last agent then initiates the *backwards* trip by sending $S = S_t$ and r to a randomly selected agent in S_t.

3. **Backwards Trip.** In the *backwards* trip, the receiving agent removes the random number y from r that it added to it in the *forwards* trip. The agent then removes itself from S and sends the updated S and r to the agent in S that it trusts the most (preferably a different agent than before). The *backwards* trip continues in this manner until the last agent in S updates r. The last agent then sends r to q.

4. **Result.** The value of r that q receives is the sum of the local feedback values of all agents in S_t. Agent q computes $r_{t,\psi} = r/n$.

The general ideas behind the correctness and privacy preservation properties of the protocol are summarized below. Detailed security analysis of the protocol is provided in Section 4.4.4. The innovations in the protocol are discussed in Section 4.4.6.

Correctness. In the forwards trip, each source agent adds its feedback value and a random number. Thus at the end of the forwards trip the sum comprises of: 1) the sum of all feedback values, and 2) the sum of all random numbers. In the backwards trip, each source agent subtracts the random number that it added in the forwards trip. Thus the final sum is the sum of all feedback values, which leads to the correct reputation value of the target agent.

Privacy Preservation. In the forwards trip, the feedback value of an agent is protected by data perturbation due to the secret random number that it adds itself. In the backwards trip the feedback value is protected by data perturbation due to the sum of the feedback values of all the other honest agents added to it. As long as there are $\Upsilon + 1$ honest source agents in the protocol, data perturbation will hold for each honest agent's feedback value in the backwards trip. The only way for the adversary to learn the feedback of an agent $\boldsymbol{a}$ is if the two agents that $\boldsymbol{a}$ selects as trustworthy turn out to be dishonest. However, the probability of that is low since the trustworthy agents are selected by $\boldsymbol{a}$ himself.

4.4.3 Protocol Specification

The protocol is specified in Figure 4.3. It is given as a collection of events and associated actions for an agent $\boldsymbol{a}$. The protocol is uniform for every participant. Table 4.3 describes the functions used in the protocol.

4.4.4 Security

Correctness

Theorem 5. *In the protocol Semi-Honest-Round-Trip (Figure 4.3), agent q learns the correct reputation of agent t in the context ψ, under the semi-honest adversarial model.*

Proof. The semi-honest adversarial model implies that the protocol is followed according to the specification and that the messages are not tampered with.

When the querying agent initiates the query, the variable $r = 0$. In the *forwards* trip, the tuple $(\text{FORWARDS}, q, t, s, \psi, r, S)$ arrives once at each agent $a \in S_t$, who adds the values of its l_{at} and $y_{(q,t,s)}$ to it. When the tuple arrives at the first agent in the *backwards* trip, each $a \in S_t$ has added the values of its l_{at} and $y_{(q,t,s)}$ to r. For the purpose of this proof, let's refer to the $y_{(q,t,s)}$ value

Table 4.3: Description of the functions used in Semi-Honest-Round-Trip.

Function	**Description**
random_element(S)	Returns a random element from the set S.
timestamp()	Returns current time.
random(α,β)	Returns a random number uniformly distributed over the interval $[\alpha, \beta]$, where α and β are constants.
trustworthy($\boldsymbol{a}$, S)	Returns an agent $\boldsymbol{b}$ from the set S such that $P(perform(a, b, \rho) = true) \geq 0 \wedge \forall v \in S - \{b\}$, $P(perform(a, b, \rho) = true) \geq P(perform(a, v, \rho) = true)$. If two or more agents meet this criteria, then one of those agents is selected at random. If none of the agents meet this criteria, then an agent is selected at random from S. Intuitively, function *trustworthy* returns an agent $\boldsymbol{b}$ that the agent $\boldsymbol{a}$ trusts the most to preserve its privacy (out of all the agents in the set S).

Protocol: Semi-Honest-Round-Trip

Participants: Agents: q, t, $S_t \equiv S_{t,\psi} = \{a_1 \dots a_n\}$. Agents q, t, and a subset of $S_{t,\psi}$ of size $m < n$ are considered to be dishonest, however, q wishes to learn the correct output. $n \geq 3$.

Input: Each source agent $\boldsymbol{a}$ has a private input $l_{at} \equiv P(perform(a, t, \psi) = true)$.

Output: Agent q learns $r_{t,\psi}$, the reputation of agent t in the context ψ.

Setup: Each agent $\boldsymbol{a}$ maintains $S_a \equiv S_{a,\psi}$, the set of its source agents in the context ψ.

Events and Associated Actions (for an Agent $\boldsymbol{a}$):

need arises to determine $r_{t,\psi}$

▷ initiate query
1 send tuple (REQUEST_FOR_SOURCES, ψ) to t
2 receive tuple (SOURCES, ψ, S_t) from t
3 $a_{(f,out)} \leftarrow$ random_element(S_t)
4 $q \leftarrow a$
5 $s \leftarrow$ timestamp()
6 $r \leftarrow 0$
7 send tuple (FORWARDS, q, t, s, ψ, r, S_t) to $a_{(f,out)}$

tuple (REQUEST_FOR_SOURCES, ψ) **received from agent** q
1 send tuple (SOURCES, ψ, S_a) to q

tuple (FORWARDS, q, t, s, ψ, r, S) **received from agent** $a_{(f,in)}$
1 $r_{(f,in)} \leftarrow r$
2 $y_{(q,t,s)} \leftarrow \text{random}(0, \Upsilon)$
3 $r_{(f,out)} \leftarrow r_{(f,in)} + l_{at} + y_{(q,t,s)}$
4 $S_{(f,in)} \leftarrow S$
5 $S_{(f,out)} \leftarrow S_{(f,in)} - \{a\}$
6 **if** $|S_{(f,out)}| > 0$
7 **then** $a_{(f,out)} \leftarrow \text{trustworthy}(a, S_{(f,out)})$
8 $a_{(q,t,s)} \leftarrow a_{(f,out)}$
9 send tuple (FORWARDS, $q, t, s, \psi, r_{(f,out)}, S_{(f,out)}$) to $a_{(f,out)}$
10 **else** send tuple (REQUEST_FOR_SOURCES, ψ) to t
11 receive tuple (SOURCES, ψ, S_t) from t
12 $a_{(f,out)} \leftarrow \text{trustworthy}(a, S_t)$
13 $a_{(q,t,s)} \leftarrow a_{(f,out)}$
14 send tuple (BACKWARDS, $q, t, s, r_{(f,out)}, S_t$) to $a_{(f,out)}$
15 store $y_{(q,t,s)}$ and $a_{(q,t,s)}$

Figure 4.3: Protocol: Semi-Honest-Round-Trip

Protocol: Semi-Honest-Round-Trip (contd.)

tuple $(\text{BACKWARDS}, q, t, s, r, S)$ **received from agent** $a_{(b,in)}$

1 $r_{(b,in)} \leftarrow r$
2 $r_{(b,out)} \leftarrow r_{(b,in)} - y_{(q,t,s)}$
3 $S_{(b,in)} \leftarrow S$
4 $S_{(b,out)} \leftarrow S_{(b,in)} - \{a\}$
5 **if** $S_{(b,out)} \neq \phi$ *and* $|S_{(b,out)} - \{a_{(q,t,s)}\}| > 0$
6 **then** $a_{(b,out)} \leftarrow \text{trustworthy}(a, S_{(b,out)} - \{a_{(q,t,s)}\})$
7 send tuple $(\text{BACKWARDS}, q, t, s, r_{(b,out)}, S_{(b,out)})$ to $a_{(b,out)}$
8 **else if** $|S_{(b,out)}| > 0$
9 **then** $a_{(b,out)} \leftarrow \text{trustworthy}(a, S_{(b,out)})$
10 send tuple $(\text{BACKWARDS}, q, t, s, r_{(b,out)}, S_{(b,out)})$ to $a_{(b,out)}$
11 **else** $a_{(b,out)} \leftarrow q$
12 send tuple $(\text{RESULT}, q, t, s, r_{(b,out)})$ to $a_{(b,out)}$
13 discard $y_{(q,t,s)}$ and $a_{(q,t,s)}$

tuple $(\text{RESULT}, q, t, s, r)$ **received from agent** $a_{(b,in)}$

1 $r_{t,\psi} \leftarrow r/n$

Figure 4.4: Protocol: Semi-Honest-Round-Trip (contd.)

of an agent $\boldsymbol{a}$ as y_a. Then the value of r when it reaches the first agent in the backwards trip is $r = \sum_{a \in S_t}(l_{at} + y_a)$.

In the *backwards* trip, the tuple $(\text{BACKWARDS}, q, t, s, \psi, r, S)$ arrives once at each agent. Each of those n agents, $a \in S_t$, subtracts y_a from r. Thus when $(\text{RESULT}, q, t, s, \psi, r, S)$ arrives at q, $r = \sum_{a \in S_t} l_{at}$. q computes $r_t = r/n$, which is the correct reputation of agent t in the context ψ (Equation 3.2). Thus agent q learns the correct reputation of agent t in the context ψ, under the semi-honest adversarial model. □

Privacy

Theorem 6. *Let's assume that in the Semi-Honest-Ideal and the Semi-Honest-Round-Trip protocols, there are $h+1$ honest source agents, such that $h \geq \Upsilon$. Moreover, assume that for each agent $a \in S_t$ in the Semi-Honest-Round-Trip protocol, $P(perform(a, a_{(f,out)}, \rho) = false) \times P(perform(a, a_{(b,out)}, \rho) = false)$ is low. Then Semi-Honest-Round-Trip is a real privacy preserving protocol under the semi-honest model.*

Proof. The semi-honest adversarial model implies that the protocol is followed according to the specification, and that the communication channels are not wire-tapped and not tampered with.

Let's consider a source agent $\boldsymbol{a}$. In the forwards trip, $\boldsymbol{a}$ receives $r_{(f,in)}$ from $a_{(f,in)}$, computes $r_{(f,out)} = r_{(f,in)} + l_{at} + y_{(q,t,s)}$, and sends the $r_{(f,out)}$ to $a_{(f,out)}$. At this point, if $a_{(f,in)}$ and $a_{(f,out)}$ are dishonest, they can learn $l_{at} + y_{(q,t,s)}$ as follows ($c_1, c_2, \ldots$ are constants):

$a_{(f,in)}$ has the following information:

$$r_{(f,in)} = c_1 \tag{4.1}$$

$a_{(f,out)}$ knows:

$$r_{(f,in)} + l_{at} + y_{(q,t,s)} = c_2 \tag{4.2}$$

From equations 4.1 and 4.2:

$$l_{at} + y_{(q,t,s)} = c_2 - c_1 = c_3 \tag{4.3}$$

However, equation 4.3 does not reveal any information about l_{at}, with high probability, due to data perturbation (Definition 22). Thus privacy is preserved by the protocol up to this step.

The data perturbation, due to the addition of $y_{(q,t,s)}$, protects l_{at} until agent $\boldsymbol{a}$ computes $r_{(b,out)} = r_{(b,in)} - y_{(q,t,s)}$ in the backwards trip and sends $r_{(b,out)}$ to $a_{(b,out)}$. However, at this point there are feedback values of h other honest agents that have been added to l_{at}. The sum of those feedback values lies in the interval $[0, h]$, where $h \geq \Upsilon$. The adversary has no knowledge of those feedback values. Thus from this point onwards, the sum of the feedback values of the other honest agents provides the data perturbation for l_{at}. Thus privacy is preserved by the protocol (except against a collusion of $a_{(f,in)}$, $a_{(f,out)}$, $a_{(b,in)}$, and $a_{(b,out)}$, as we discuss below).

At this point, if $a_{(b,in)}$ and $a_{(b,out)}$ are dishonest, they can learn $y_{(q,t,s)}$ as follows:

$a_{(b,in)}$ has the following information:

$$r_{(b,in)} = c_4 \tag{4.4}$$

$a_{(b,out)}$ knows:

$$r_{(b,in)} - y_{(q,t,s)} = c_5 \tag{4.5}$$

From equations 4.4, and 4.5:

$$y_{(q,t,s)} = c_4 - c_5 = c_6 \tag{4.6}$$

However, $y_{(q,t,s)}$ does not give any information about l_{at}. The only option at this point for $a_{(b,in)}$ and $a_{(b,out)}$ to learn l_{at} is to collude with $a_{(f,in)}$ and $a_{(f,out)}$ from the previous trip. If $a_{(f,in)}$, $a_{(f,out)}$, $a_{(b,in)}$, and $a_{(b,out)}$ all collude, then equations 4.3 and 4.6 give:

$$l_{at} = c_3 - c_6 = c_7$$

The probability that $a_{(f,in)}$, $a_{(f,out)}$, $a_{(b,in)}$, and $a_{(b,out)}$ will collude is given as follows: $P(a_{(f,in)} \in \mathbb{M}) \times P(a_{(f,out)} \in \mathbb{M}) \times P(a_{(b,in)} \in \mathbb{M}) \times P(a_{(b,out)} \in \mathbb{M})$.

Since $\boldsymbol{a}$ has no control over who $a_{(f,in)}$ and $a_{(b,in)}$ are, we assume that they are in $\mathbb{M}$ and thus $P(a_{(f,in)} \in \mathbb{M}) = 1$ and $P(a_{(b,in)} \in \mathbb{M}) = 1$. The probability that the four agents will collude can be revised as: $P(a_{(f,out)} \in \mathbb{M}) \times P(a_{(b,out)} \in \mathbb{M})$, or equivalently: $P(perform(a, a_{(f,out)}, \rho) = false) \times P(perform(a, a_{(b,out)}, \rho) = false)$.

Since we assume that $P(perform(a, a_{(f,out)}, \rho) = false) \times P(perform(a, a_{(b,out)}, \rho) = false)$ is low, privacy is preserved by the protocol.

The worst case scenario after this point is that all remaining agents in the backwards trip are dishonest. Even if that is the case, the adversary does not learn any more information than in the ideal protocol. Thus privacy is preserved by the protocol.

The protocol Semi-Honest-Round-Trip is a real privacy preserving reputation protocol (Definition 20) under the semi-honest model, since: 1) Semi-Honest-Round-Trip has the same parameters as Semi-Honest-Ideal (except the TTP), with the assumption that there are $h+1$ honest source agents in both, such that $h \geq \Upsilon$, and 2) the adversary learns no more information about the private input of an agent $\boldsymbol{a}$ in Semi-Honest-Round-Trip than it can learn in Semi-Honest-Ideal, with high probability, under the semi-honest adversarial model. □

We make the following two assumptions to show that the protocol Semi-Honest-Round-Trip is a real privacy preserving protocol: 1) there are $h+1$ honest source agents in the protocol, such that $h \geq \Upsilon$; and 2) $P(perform(a, a_{(f,out)}, \rho) = false) \times P(perform(a, a_{(b,out)}, \rho) = false)$ is low for each agent $a \in S_t$.

The assumption that there are $h+1$ honest source agents, such that $h \geq \Upsilon$, implies that the number of dishonest agents is: $m = n - (h+1)$, or $m < n - (\Upsilon + 1)$. Since Υ is positive, this assumption conforms with the prior more general assumption under both protocols (ideal and real) that the number of dishonest agents is $m < n$. In Section 4.5.2, we propose an extension to the protocol to cater for scenarios where the presence of $h+1$ honest source agents cannot be assumed.

In Section 4.5.1, we present an extension to the protocol that allows an agent to abstain from providing feedback. This option preserves the privacy of an agent $\boldsymbol{a}$ in the case when $P(perform(a, a_{(f,out)}, \rho) = false) \times P(perform(a, a_{(b,out)}, \rho) = false)$ is not low. In Experiment 1 (Section 4.8.1), we study the percentage of instances of source agents in a real web of trust (Advogato.org) for whom this assumption does hold true.

4.4.5 Complexity

Table 4.4: Protocol Semi-Honest-Round-Trip – Complexity.

Tuple	**Occurrences**	**IDs**	**Numbers**
REQUEST_FOR_SOURCES	2		
SOURCES	2	$2n$	
FORWARDS	n	$2n + \frac{n(n+1)}{2}$	$n(1) = n$
BACKWARDS	n	$2n + \frac{n(n+1)}{2}$	$n(1) = n$
RESULT	1	2	1
Total	$2n + 3$	$n^2 + 7n + 2$	$2n + 1$
Complexity	$O(n)$	$O(n^2)$	$O(n)$

The protocol requires $2n + 3$ messages to be exchanged (complexity: $O(n)$).

In terms of bandwidth used, the protocol requires transmission of the following amount of information: $n^2 + 7n + 2$ agent IDs (complexity: $O(n^2)$), and $2n + 1$ numbers (complexity: $O(n)$).

4.4.6 Discussion

The Round-Trip protocol has the following innovations:

1. An agent $\boldsymbol{a}$ is aware of its trust relationships with fellow source agents in the context of preserving privacy / being honest.

2. An agent himself selects the agents that are critical for preserving its privacy. The agent $\boldsymbol{a}$ can select the agents that it trusts the most, in order to maximize the probability that its privacy will be preserved. As a consequence, the order of the agents in the protocol is not pre-determined by the querying agent. Thus, unlike in the secure sum protocol, the querying agent cannot set up a source agent for its privacy to be compromised.

3. Two trips disable the predecessor and the successor of only a single trip from being able to breach an agent's privacy.

4. An agent is able to quantify the privacy guarantee as subjective probability, before it relies on trustworthy agents to preserve its privacy.

As opposed to the secure sum protocol, the Round-Trip protocol is secure under the full semi-honest model. Whereas the message complexity of both protocols is the same, that is, linear in terms of the number of messages, and quadratic in terms of the number of IDs exchanged. The extension presented in the following section, which allows agents to abstain, ensures that their privacy is preserved even when trustworthy agents are not available. Please see Section 4.8 for a discussion of the experimental results.

A drawback of the Round-Trip protocol is as follows: The first agent in the *forwards* or *backwards* trip can choose its trustworthy agent from the full pool of all source agents. However, subsequent agents in the trip have less and less choice since agents who have already been chosen in that trip cannot be chosen again. The k-Shares protocol (presented in Section 4.6) fixes this drawback and offers each agent the full pool of fellow agents to choose from.

Another drawback of the Round-Trip protocol is that it requires at least $\Upsilon + 1$ honest source agents. The k-Shares protocol eliminates this requirement as well.

As compared to the k-Shares protocol, the Round-Trip protocol places less communication and processing overhead on the querying agent. The Round-Trip protocol may be used instead of k-Shares if this property is desired.

4.5 Extensions to the Round-Trip Reputation Protocol

4.5.1 The Option to Abstain

The privacy of the Round-Trip Reputation Protocol presented in the previous section depends on the assumption that $P(perform(a, a_{(f,out)}, \rho) = false) \times P(perform(a, a_{(b,out)}, \rho) = false)$ is low for each agent $a \in S_t$. In other words, the probability that the successors of an agent in the *forwards* trip and the *backwards* trip are both dishonest is low for each agent. In this section, we extend the protocol such that an agent $\boldsymbol{a}$ has the option to abstain from contributing his feedback when this assumption does not hold. The extensions are made to the *backwards* trip and the *result* steps of the protocol.

Up to Perfect Privacy

Pavlov et al. [105] argue that it is impossible to guarantee perfect privacy for an honest feedback provider in a decentralized additive reputation protocol. The argument is that a dishonest agent may deterministically create a set of n feedback providers, with $n-1$ dishonest agents and the one honest agent under attack. Given the inputs of the $n-1$ dishonest agents and the output (the reputation score), the secret feedback of the honest agent is easily obtained.

However, we argue that perfect privacy is in fact possible for an honest feedback provider in a decentralized additive reputation protocol. Our observation is that the privacy of the honest agent is perfectly preserved if the agent abstains

from providing its feedback. This option is not considered by Pavlov et al. In their protocols, agents are not able to quantify the privacy guarantee, before they rely on fellow agents to protect their privacy. Thus an agent is unable to evaluate the risk to its privacy. On the other hand, in our protocols, an agent knows the privacy guarantee, before it leaves it up to others to preserve its privacy. If the guarantee quantified as subjective probability is insufficient, the agent may abstain, thus perfectly preserving its privacy.

We note that allowing agents to abstain from providing feedback has an effect on the correctness of the reputation values computed by the protocol. In Section 4.5.1, we define two metrics to measure the correctness of the reputation values computed when agents are allowed to abstain: 1) Resolution, and 2) Disparity.

Protocol Outline

The extensions are described below.

- **Backwards Trip.** The first step now for an agent $\boldsymbol{a}$ in the *backwards* trip is to select the agent in the set $S-\{a\}$ that it trusts the most to preserve its privacy. At this stage, agent $\boldsymbol{a}$ knows the identities of $a_{(f,out)}$ and $a_{(b,out)}$, who are its successors in the *forwards* trip and *backwards* trip respectively. Hence, agent $\boldsymbol{a}$ can compute the probability $P(perform(a, a_{(f,out)}, \rho) = false) \times P(perform(a, a_{(b,out)}, \rho) = false)$. If agent $\boldsymbol{a}$ finds that this probability is sufficiently low, he can proceed normally. Otherwise, the agent $\boldsymbol{a}$ can opt to abstain and proceed as follows: Agent $\boldsymbol{a}$ removes y as well as l_{at} from r. This is in contrast to the original protocol in which agent $\boldsymbol{a}$ removes only y from r. Agent $\boldsymbol{a}$ then adds itself to a set B, which is the set of all agents who have abstained. Finally, it sends the updated S, r, and B to its successor. The *backwards* trip continues in this manner until the last agent in S has acted. The last agent then sends r to q.
- **Result.** The value of r that q receives in this extended version is the sum of the local feedback values of the agents that did not abstain, that is, the agents $S_t - B$. Agent q computes $r'_{t,\psi} = r/|S_t - B|$.

Protocol Specification

The modifications to the Semi-Honest-Round-Trip protocol are given in Figure 4.5.

Security: Correctness

We define two metrics to measure the correctness of the reputation value computed by the extended protocol: 1) Resolution, and 2) Disparity.

Definition 24. ***Resolution.*** *Let's consider that reputation is computed as* $r'_{t,\psi} = \sum_{a \in S_t - B} l_{at}/|S_t - B|$, *instead of* $r_{t,\psi} = \sum_{a \in S_t} l_{at}/|S_t|$. *Then the resolution of the reputation is defined as* $res(r'_{t,\psi}) = |S_t - B|/|S_t|\%$. *That is,*

Protocol: Semi-Honest-Round-Trip-Abstain

Modified Events and Associated Actions:

tuple $(\text{FORWARDS}, q, t, s, \psi, r, S)$ **received from agent** $a_{(f,in)}$

▷ lines 1 through 13 are the same as before

14 $B \leftarrow \phi$

15 send tuple $(\text{BACKWARDS}, q, t, s, r_{(f,out)}, S_t, B)$ to $a_{(f,out)}$

16 store $y_{(q,t,s)}$ and $a_{(q,t,s)}$

tuple $(\text{BACKWARDS}, q, t, s, r, S, B)$ **received from agent** $a_{(b,in)}$

1 $r_{(b,in)} \leftarrow r$

2 $S_{(b,in)} \leftarrow S$

3 $S_{(b,out)} \leftarrow S_{(b,in)} - \{a\}$

4 **if** $S_{(b,out)} \neq \phi$ *and* $|S_{(b,out)} - \{a_{(q,t,s)}\}| > 0$

5 **then** $a_{(b,out)} \leftarrow \text{trustworthy}(a, S_{(b,out)} - \{a_{(q,t,s)}\})$

6 **else if** $|S_{(b,out)}| > 0$

7 **then** $a_{(b,out)} \leftarrow \text{trustworthy}(a, S_{(b,out)})$

8 **else** $a_{(b,out)} \leftarrow q$

9 **if** $P(perform(a, a_{(q,t,s)}, \rho) = false) \times$
$P(perform(a, a_{(b,out)}, \rho) = false)$ is low

10 **then** $r_{(b,out)} \leftarrow r_{(b,in)} - y_{(q,t,s)}$

11 $B_{out} \leftarrow B$

12 **else** $r_{(b,out)} \leftarrow r_{(b,in)} - y_{(q,t,s)} - l_{at}$

13 $B_{out} \leftarrow B \cup \{a\}$

14 **if** $a_{(b,out)} = q$

15 **then** send tuple $(\text{RESULT}, q, t, s, r_{(b,out)}, B_{out})$ to $a_{(b,out)}$

16 **else** send tuple $(\text{BACKWARDS}, q, t, s, r_{(b,out)}, S_{(b,out)}, B_{out})$ to $a_{(b,out)}$

17 discard $y_{(q,t,s)}$ and $a_{(q,t,s)}$

tuple $(\text{RESULT}, q, t, s, r, B)$ **received from agent** $a_{(b,in)}$

1 $r'_{t,\psi} \leftarrow r/|S_t - B|$

Figure 4.5: Protocol: Semi-Honest-Round-Trip-Abstain

the percentage of the total number of source agents that contributed towards the computation of the reputation value.

A resolution of 0% means that no source agents contributed, whereas a resolution of 100% signifies that all source agents contributed.

Definition 25. ***Disparity.*** *Let's consider that reputation is computed as* $r'_{t,\psi} = \sum_{a \in S_t - B} l_{at}/|S_t - B|$*, instead of* $r_{t,\psi} = \sum_{a \in S_t} l_{at}/|S_t|$*. Then the disparity*

of the reputation value is defined as $disp(r_{t,\psi}) = |r_{t,\psi} - r'_{t,\psi}|$. *That is, the absolute difference between the reputation computed with all source agents and the reputation computed with only the source agents in* $S_t - B$.

The disparity would range from 0 to 1. The lower the disparity, the more accurate is the reputation. A disparity of 0 means that a reputation value computed with less than all source agents is exactly the same as it would be if computed with all source agents.

In Experiment 2 (Section 4.8.1), we determine the resolution and the disparity of the reputation values computed by the protocol Semi-Honest-Round-Trip-Abstain in the real web of trust of Advogato.org.

4.5.2 Seed Agents

The privacy of the Round-Trip Reputation Protocol also depends on the assumption that there are $h + 1$ honest source agents in the protocol, where $h \geq \Upsilon$. In this section, we extend the protocol such that the protocol preserves privacy even if this assumption does not hold. The extensions are made to the *forwards* trip and the *backwards* trip of the protocol. Moreover, an additional *seed* step is added to the protocol.

Tools and Techniques: Seed Agents

The idea of seed agents is inspired by the reputation systems EigenTrust [80] and Advogato [87]. Seed agents are known trustworthy agents in the environment. However, please note that the seed agents are not TTPs. In this protocol extension, an agent does not rely completely on a seed agent for its privacy. If a seed is dishonest, it would still have to collude with at minimum the successor of $\boldsymbol{a}$ in the *backwards* trip to breach $\boldsymbol{a}$'s privacy.

Protocol Outline

The extensions are described below.

- **Forwards Trip.** The last agent in the *forwards* trip sends the updated S and r to a seed agent. This is in contrast to the previous versions, in which the last agent in the *forwards* trip directly initiated the *backwards* trip.
- **Seed.** The seed agent generates random numbers $x_1 \ldots x_{n-1}$ and then selects x_n such that the sum of the n numbers is equal to 0. It then sends each x_i to each $a_i \in S_t$, where $i \in \{1 \ldots n\}$. The seed then initiates the *backwards* trip by sending $S = S_t$ and r to a randomly selected agent in S_t.
- **Backwards Trip.** As an additional step, each agent a_i in the *backwards* trip adds the x_i value received from the seed to the sum r.

Protocol Specification

The modifications to the Semi-Honest-Round-Trip-Abstain protocol are given in Figure 4.6.

4.6 The k-Shares Reputation Protocol

In this section we present our k-shares protocol, which is also a real privacy preserving reputation protocol under the semi-honest model. The k-Shares protocol allows each feedback provider to choose its trustworthy agents from the complete set of source agents. In contrast, the Round-Trip offers most feedback providers only a limited choice of source agents.

4.6.1 Tools and Techniques: Secret Sharing

Secret sharing is a technique that allows a secret to be shared among n players, such that the secret can be learned only when a minimum of t of the n players cooperate to learn the secret. Any coalition of less than t players is unable to learn any information about the secret. We refer the reader to [123] for a discussion on secret sharing.

In the k-shares protocol we use the simple case of secret sharing where $t = n$.

4.6.2 Protocol Outline

The important steps of the protocol are outlined below.

1. **Initiation.** The protocol is initiated by a querying agent q to determine the reputation $r_{t,\psi}$ of a target agent t. Agent q retrieves $S_t \equiv S_{t,\psi}$, the set of source agents of agent t in the context ψ. Agent q then sends S_t to each agent $a \in S_t$.

2. **Select Trustworthy Agents.** Each agent $a \in S_t$ selects up to k other agents in S_t. Let's refer to these agents selected by $\boldsymbol{a}$ as the set $U_a = \{u_{a,1} \ldots u_{a,k_a}\}$, where $1 \leq k_a \leq k$. Agent $\boldsymbol{a}$ selects these agents such that: $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$ is low. That is, the probability that all of the selected agents will collude to break agent $\boldsymbol{a}$'s privacy is low.

3. **Prepare Shares.** Agent $\boldsymbol{a}$ then prepares $k_a + 1$ shares of its secret feedback value l_{at}. The shares, given as: $x_{a,1} \ldots x_{a,k_a+1}$, are prepared as follows: The first k_a shares are random numbers uniformly distributed over a large interval. The last share is selected such that: $\sum_{i=1}^{k_a+1} x_{a,i} = l_{at}$. That is, the sum of the shares is equal to the feedback value. Since each of the $k_a + 1$ shares is a number uniformly distributed over a large interval, no information about the secret can be learnt unless all of the shares are known.

Protocol: Semi-Honest-Round-Trip-Seeds

Setup: In this extended protocol, each agent $\boldsymbol{a}$ additionally knows the set D, which is the set of seed agents.

Modified Events and Associated Actions:

tuple $(\text{FORWARDS}, q, t, s, \psi, r, S)$ **received from agent** $a_{(f,in)}$

▷ lines 1 through 9 are the same as before
10 **else** $a_{(f,out)} \leftarrow \text{random\_element}(D)$
11 $a_{(q,t,s)} \leftarrow a_{(f,out)}$
12 send tuple $(\text{SEED}, q, t, s, \psi, r)$ to $a_{(f,out)}$
13 store $y_{(q,t,s)}$ and $a_{(q,t,s)}$

tuple $(\text{SEED}, q, t, s, \psi, r)$ **received from agent** $a_{(f,in)}$

1 **for** $i \leftarrow 1$ **to** $n-1$
2 **do** $x_i \leftarrow \text{random}(-X, X)$
3 $x_n \leftarrow -\sum_{i=1}^{n-1} x_i$
4 **for** each agent $a_i \in S_t = \{a_1 \ldots a_n\}$
5 **do** send tuple $(\text{PARTX}, q, t, s, \psi, x_i)$ to agent a_i
6 $B \leftarrow \phi$
7 send tuple $(\text{REQUEST\_FOR\_SOURCES}, \psi)$ to t
8 receive tuple $(\text{SOURCES}, \psi, S_t)$ from t
9 $a_{(b,out)} \leftarrow \text{random\_element}(S_t)$
10 send tuple $(\text{BACKWARDS}, q, t, s, \psi, r, S_t, B)$ to $a_{(b,out)}$

tuple $(\text{PARTX}, q, t, s, \psi, x)$ **received from agent** d

1 $x_{(q,t,s)} \leftarrow x$
2 store $x_{(q,t,s)}$

tuple $(\text{BACKWARDS}, q, t, s, \psi, r, S, B)$ **received from agent** $a_{(b,in)}$

▷ lines 1 through 9 are the same as before
10 **then** $r_{(b,out)} \leftarrow r_{(b,in)} - y_{(q,t,s)} + x_{(q,t,s)}$
11 $B_{out} \leftarrow B$
12 **else** $r_{(b,out)} \leftarrow r_{(b,in)} - y_{(q,t,s)} - l_{at} + x_{(q,t,s)}$
13 $B_{out} \leftarrow B \cup \{a\}$
14 **if** $a_{(b,out)} = q$
15 **then** send tuple $(\text{RESULT}, q, t, s, \psi, r_{(b,out)}, B_{out})$ to $a_{(b,out)}$
16 **else** send tuple $(\text{BACKWARDS}, q, t, s, \psi, r_{(b,out)}, S_{(b,out)}, B_{out})$ to $a_{(b,out)}$
17 discard $y_{(q,t,s)}$, $a_{(q,t,s)}$, and $x_{(q,t,s)}$

Figure 4.6: Protocol: Semi-Honest-Round-Trip-Seeds

4. **Send Shares.** Agent $\boldsymbol{a}$ sends the set $U_a = \{u_{a,1} \dots u_{a,k_a}\}$ to agent q. Agent $\boldsymbol{a}$ sends $x_{a,i}$ to agent $u_{a,i}$, where $i \in \{1 \dots k_a\}$.

5. **Receive Shares.** Agent q receives U_a from each agent $a \in S_t$. Then for each agent $\boldsymbol{a}$, agent q: 1) compiles the list of agents from whom $\boldsymbol{a}$ should expect to receive shares, and 2) sends this list to agent $\boldsymbol{a}$. Agent $\boldsymbol{a}$ then proceeds to receive shares from the agents on the list provided by q.

6. **Compute Sums.** Agent $\boldsymbol{a}$ computes σ_a, the sum of all shares received and its own final share x_{a,k_a+1}. Agent $\boldsymbol{a}$ sends the sum σ_a to q.

7. **Compute Reputation.** Agent q receives the sum σ_a from each agent $a \in S_t$. q computes $r_{t,\psi} = (\sum_{a \in S_t} \sigma_a)/n$.

The correctness of the protocol is dependent on the assumption that the querying agent will receive the sum of all shares of all agents. This is not straightforward since each agent independently sends out an arbitrary number of shares. The protocol ensures that each agent receives all shares by tasking the querying agent to compile and distribute lists of the senders.

Regarding privacy preservation, the privacy of an agent $\boldsymbol{a}$ cannot be breached unless all agents whom $\boldsymbol{a}$ sent shares to are dishonest. This is due to the fact that the shares are random numbers uniformly distributed over a large interval and thus do not reveal any information unless all of them are known.

4.6.3 Protocol Specification

The protocol is specified in Figure 4.7. Table 4.5 describes the functions used in the protocol.

Table 4.5: Description of the functions used in Semi-Honest-k-Shares.

Function	**Description**
timestamp()	Returns current time.
random(α,β)	Returns a random number uniformly distributed over the interval $[\alpha, \beta]$, where α and β are constants.
set_of_trustworthy($\boldsymbol{a}$, S)	Returns a set of agents $U_a = \{u_{a,1} \dots u_{a,k_a}\}$, where $1 \leq k_a \leq k$, and $U_a \subseteq S$. The set U_a is selected such that: $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$ is low, with the minimum possible k_a.

Protocol: Semi-Honest-k-Shares

Participants: Agents: q, t, $S_t \equiv S_{t,\psi} = \{a_1 \dots a_n\}$. Agents q, t, and a subset of $S_{t,\psi}$ of size $m < n$ are considered to be dishonest, however, q wishes to learn the correct output. $n \geq 3$.

Input: Each source agent $\boldsymbol{a}$ has a private input $l_{at} \equiv P(perform(a, t, \psi) = true)$.

Output: Agent q learns $r_{t,\psi}$, the reputation of agent t in the context ψ.

Setup: Each agent $\boldsymbol{a}$ maintains $S_a \equiv S_{a,\psi}$, the set of its source agents in the context ψ.

Events and Associated Actions (for an Agent $\boldsymbol{a}$):

need arises to determine $r_{t,\psi}$

▷ initiate query
1 send tuple (REQUEST_FOR_SOURCES, ψ) to t
2 receive tuple (SOURCES, ψ, S_t) from t
3 **for** each agent $v \in S_t$
4 **do** $J_v \leftarrow \phi$ ▷ set of agents who have chosen agent v as trustworthy
5 $S'_t \leftarrow S_t$ ▷ intermediate set of source agents
6 $r \leftarrow 0$ ▷ intermediate sum of feedback
7 $q \leftarrow a$ ▷ the current agent is the querying agent
8 $s \leftarrow$ timestamp()
9 send tuple (PREP, q, t, s, S_t) to each agent $v \in S_t$

tuple (REQUEST_FOR_SOURCES, ψ) **received from agent** q

1 send tuple (SOURCES, ψ, S_a) to q

tuple (PREP, q, t, s, S_t) **received from agent** q

1 $I \leftarrow \phi$ ▷ set of agents from whom the current agent has received shares
2 $J \leftarrow \phi$ ▷ set of agents who have chosen the current agent as trustworthy
3 $\sigma_a \leftarrow 0$ ▷ current agent's local sum
4 $U_a \leftarrow$ set_of_trustworthy($\boldsymbol{a}$, $S_t - \{a\}$)
5 $k_a \leftarrow |U_a|$
6 **for** $i \leftarrow 1$ **to** k_a ▷ prepare shares
7 **do** $x_{a,i} \leftarrow$ random$(-X, X)$
8 $x_{a,k_a+1} \leftarrow l_{at} - \sum_{i=1}^{k_a} x_{a,i}$ ▷ prepare final share
9 send tuple (RECIPIENTS, q, t, s, U_a) to agent q
10 **for** each agent $u_{a,i} \in U_a = \{u_{a,1} \dots u_{a,k_a}\}$
11 **do** send tuple (SHARE, $q, t, s, x_{a,i}$) to agent $u_{a,i}$

Figure 4.7: Protocol: Semi-Honest-k-Shares

Protocol: Semi-Honest-k-Shares (contd.)

tuple $(\text{RECIPIENTS}, q, t, s, U_v)$ **received from an agent** $v \in S_t$

```
1  for each agent u ∈ U_v
2        do J_u ← J_u ∪ {v}
3  S'_t ← S'_t − {v}
4  if S'_t = φ ▷ if RECIPIENTS has been received from all source agents
5     then S'_t ← S_t
6          for each agent w ∈ S_t
7               do send tuple (SENDERS, q, t, s, J_w) to agent w
```

tuple $(\text{SHARE}, q, t, s, x_v)$ **received from an agent** $v \in S_t$

```
1  I ← I ∪ {v}
2  σ_a ← σ_a + x_v
3  if I = J ▷ if SHARE has been received from all agents in J
4     then σ_a ← σ_a + x_{a,k_a+1} ▷ add current agent's own final share
5          send tuple (SUM, q, t, s, σ_a) to agent q
```

tuple $(\text{SENDERS}, q, t, s, J_a)$ **received from agent** q

```
1  J ← J_a ▷ set of agents who have chosen the current agent as trustworthy
2  if I = J ▷ if SHARE has been received from all agents in J
3     then σ_a ← σ_a + x_{a,k_a+1} ▷ add current agent's own final share
4          send tuple (SUM, q, t, s, σ_a) to agent q
```

tuple $(\text{SUM}, q, t, s, \sigma_v)$ **received from an agent** $v \in S_t$

```
1  S'_t ← S'_t − {v}
2  r ← r + σ_v
3  if S'_t = φ ▷ if SUM has been received from all source agents
4     then r_{t,ψ} ← r/n
```

Figure 4.8: Protocol: Semi-Honest-k-Shares (contd.)

4.6.4 Security

Correctness

Theorem 7. *In the protocol Semi-Honest-k-Shares (Figure 4.7), agent q learns the correct reputation of agent t in the context ψ, under the semi-honest adversarial model.*

Proof. The semi-honest adversarial model implies that the protocol is followed according to the specification and that the messages are not tampered with.

In the protocol, each agent $a \in S_t$ prepares the shares $x_{a,1} \dots x_{a,k_a+1}$ of its feedback value l_{at}, such that: $\sum_{j=1}^{k_a+1} x_{a,j} = l_{at}$.

The sum of the feedback values of all agents in $S_t = \{a_1 \dots a_n\}$ is given as: $\sum_{i=1}^{n} l_{a_i t}$.

From the above two statements, the sum of the feedback values of all agents in S_t can be stated as: $\sum_{i=1}^{n}(\sum_{j=1}^{k_{a_i}+1} x_{a_i,j})$. That is, the sum of all shares of all agents.

The steps of the protocol discussed in the following paragraph ensure that all shares of all agents are included and that each share is included only once in the sums $\sigma_{a_1} \dots \sigma_{a_n}$.

Each agent $a \in S_t$ provides agent q the set U_a, which is the set of agents whom $\boldsymbol{a}$ is going to send its shares. After q has received this set from all agents in S_t, it compiles and sends to each agent $\boldsymbol{a}$, the set J_a, which is the set of agents who are in the process of sending a share to agent $\boldsymbol{a}$. Thus, each agent $\boldsymbol{a}$ knows exactly which and how many agents, it will receive a share from. When agent $\boldsymbol{a}$ has received all of those shares, it sends σ_a, the sum of all shares received and its final share, to agent q. Previously, each agent $a \in S_t$ sends each of his shares $x_{a,1} \dots x_{a,k_a}$, once to only one other agent, and adds the final share x_{a,k_a+1} once to his own σ_a. It follows that the sums $\sigma_{a_1} \dots \sigma_{a_n}$ include all shares of all agents and that they include each share only once.

The final value of r in the protocol is: $r = \sum_{i=1}^{n} \sigma_{a_i} = \sum_{i=1}^{n}(\sum_{j=1}^{k_{a_i}+1} x_{a_i,j}) = \sum_{i=1}^{n} l_{a_i t}$. Thus when q computes $r_{t,\psi} = r/n$, it is the correct reputation of agent t in the context ψ (Equation 3.2).

It follows that in the protocol Semi-Honest-k-Shares (Figure 4.7), agent q learns the correct reputation of agent t in the context ψ, under the semi-honest adversarial model. □

Privacy

Theorem 8. *Let's assume that for each agent $a \in S_t$ in the Semi-Honest-k-Shares protocol, $P(perform(a, u_{a,1}, \rho) = false) \times \dots \times P(perform(a, u_{a,k_a}, \rho) = false)$ is low. Then Semi-Honest-k-Shares is a real privacy preserving protocol under the semi-honest model.*

Proof. The semi-honest adversarial model implies that the protocol is followed according to the specification, and that the communication channels are not wire-tapped and not tampered with.

Let's consider an agent $a \in S_t$. Agent $\boldsymbol{a}$ prepares the shares $x_{a,1} \ldots x_{a,k_a+1}$ of its secret feedback value l_{at}. The first k_a shares $x_{a,1} \ldots x_{a,k_a}$ are random numbers uniformly distributed over a large interval $[-X, X]$. The final share, $x_{a,k_a+1} = l_{at} - \sum_{i=1}^{k_a} x_{a,i}$, is also a number uniformly distributed over a large interval since it is a function of the first k_a shares which are random numbers. Thus, individually each of the shares does not reveal any information about the secret feedback value l_{at}. Moreover, no information is learnt about l_{at} even if up to k_a shares are known, since their sum would be some random number uniformly distributed over a large interval. The only case in which information can be gained about l_{at} is if all k_a+1 shares are known. Then, $l_{at} = \sum_{i=1}^{k_a+1} x_{a,i}$.

We now analyze if the $k_a + 1$ shares of an agent $\boldsymbol{a}$ can be learnt by the adversary from the protocol.

Agent $\boldsymbol{a}$ sends each share $x_{a,i}$ only to agent $u_{a,i}$, where $i \in \{1 \ldots k_a\}$. Each $u_{a,i}$ then computes $\sigma_{u_{a,i}}$, which is the sum of all shares that it receives and its own final share $x_{u_{a,i},k_{u_{a,i}}+1}$. Even if agent $\boldsymbol{a}$ is the only agent to send agent $u_{a,i}$ a share, $\sigma_{u_{a,i}} = x_{a,i} + x_{u_{a,i},k_{u_{a,i}}+1}$. That is, the sum of agent $\boldsymbol{a}$'s share and agent $u_{a,i}$'s final share. $\sigma_{u_{a,i}}$ is a number uniformly distributed over a large interval. Thus, when agent $u_{a,i}$ sends this number to agent q, it is impossible for q to distinguish the individual shares from the number. Therefore, each share $x_{a,i}$ that agent $\boldsymbol{a}$ sends to agent $u_{a,i}$ will only be known to agent $u_{a,i}$. Unless, agent $u_{a,i}$ is dishonest. The probability that agent $u_{a,i}$ is dishonest, that is, it will attempt to breach agent $\boldsymbol{a}$'s privacy is given as: $P(\mathit{perform}(a, u_{a,i}, \rho) = \mathit{false})$.

To learn the first k_a shares of agent $\boldsymbol{a}$, all agents $u_{a,1} \ldots u_{a,k_a}$ would have to be dishonest. The probability of this scenario is given as: $P(\mathit{perform}(a, u_{a,1}, \rho) = \mathit{false}) \times \ldots \times P(\mathit{perform}(a, u_{a,k_a}, \rho) = \mathit{false})$.

Even in the above scenario, the adversary does not gain information about l_{at}, without the knowledge of agent $\boldsymbol{a}$'s final share x_{a,k_a+1}. However, agent $\boldsymbol{a}$ has to send $\sigma_a = x_{a,k_a+1} + \sum_{v \in J_a} x_v$, and agent $\boldsymbol{a}$ has no control over the $\sum_{v \in J_a} x_v$ portion of the equation. If $J_a = \phi$, then $\sigma_a = x_{a,k_a+1}$, which agent a must send to agent q. Therefore, we assume that agent q can learn the final share of agent $\boldsymbol{a}$.

Thus the probability that the protocol will not preserve agent $\boldsymbol{a}$'s privacy can be stated as: $P(\mathit{perform}(a, u_{a,1}, \rho) = \mathit{false}) \times \ldots \times P(\mathit{perform}(a, u_{a,k_a}, \rho) = \mathit{false})$. We assume that the agents $u_{a,1} \ldots u_{a,k_a}$ are selected such that this probability is low. Therefore, with high probability, the adversary learns no more information about l_{at} than it can learn in the ideal protocol with what it knows before the execution of the protocol and the outcome.

The protocol Semi-Honest-k-Shares is a real privacy preserving reputation protocol (Definition 20) under the semi-honest model, since: 1) Semi-Honest-k-Shares has the same parameters as Semi-Honest-Ideal (except the TTP), and 2) the adversary learns no more information about the private input of an agent $\boldsymbol{a}$ in Semi-Honest-k-Shares than it can learn in Semi-Honest-Ideal, with high probability, under the semi-honest adversarial model. □

In the k-Shares protocol, an agent must send U_a to agent q, where U_a is the set of agents whom $\boldsymbol{a}$ considers trustworthy. Although this step does not disclose

any information about agent $\boldsymbol{a}$'s private feedback, it does reveal $\boldsymbol{a}$'s preferences in terms of trustworthy agents. To counter this issue we propose the following extension to the protocol: If $|U_a| < k$, then $\boldsymbol{a}$ can add $k - |U_a|$ more agents to U_a. These additional agents should be selected randomly from the remaining source agents in S_t. The consequence of this extension is that the set U_a no longer constitutes exclusively of agents whom agent $\boldsymbol{a}$ considers trustworthy. Thus agent q cannot learn whether an agent in U_a is agent $\boldsymbol{a}$'s trustworthy agent or an agent that has been randomly selected. An adversary may attempt repeated queries to determine the agents that occur most frequently in U_a, thus revealing a's trustworthy agents. This attack can be countered if agent a selects the same set of agents for each repeated query.

4.6.5 Complexity

Table 4.6: Protocol Semi-Honest-k-Shares – Complexity.

Tuple	**Occurrences**	**IDs**	**Numbers**
REQUEST_ FOR_SOURCES	1		
SOURCES	1	n	
PREP	n	$n(n+1) = n^2 + 2n$	
RECIPIENTS	n	$n(k+2) = kn + 2n$	
SHARE	kn	$kn(2) = 2kn$	$kn(1) = kn$
SENDERS	n	$n(n) = n^2$	
SUM	n		$n(1) = n$
Total	$4n + kn + 2$	$2n^2 + 5n + 3kn$	$n + kn$
Complexity	$O(n)$, for $k \ll n$	$O(n^2)$, for $k \ll n$	$O(n)$, for $k \ll n$

The protocol requires up to $4n + kn + 2$ messages to be exchanged (complexity: $O(n)$).

In terms of bandwidth used, the protocol requires transmission of up to the following amount of information: $2n^2+5n+3kn$ agent IDs (complexity: $O(n^2)$), and $n + kn$ numbers (complexity: $O(n)$).

4.6.6 Discussion

A drawback of the Round-Trip protocol is as follows: The first agent in the *forwards* or *backwards* trip may choose its trustworthy agent from the full pool of all source agents. However, subsequent agents in the trip have less and less choice since agents who have already been chosen in that trip cannot be chosen again.

The k-Shares protocol allows each feedback provider to choose from all other $n-1$ source agents in the protocol as its trustworthy agents. This is in contrast to the Round-Trip protocol, which permits the feedback providers to choose only from those agents who have not already been chosen by the others. As a result, the k-Shares protocol can preserve the privacy of a higher percentage

of instances of source agents in a given web of trust. Our experimental results (Section 4.8) show that this percentage is over twice as high as compared to the Round-Trip protocol in the web of trust of Advogato.org.

As compared to the Round-Trip protocol, the k-Shares protocol places additional communication and processing overhead on the querying agent. This may be considered an advantage or a disadvantage. *Advantage:* The querying agent who initiates the protocol has to invest its own resources towards the execution of the protocol. This discourages agents from mounting a Denial of Service (DoS) attack by initiating queries for the purpose of tying down the resources of other agents. *Disadvantage:* It costs the querying agent more resources to learn the reputation of a target agent.

The k-shares protocol requires up to $4n + kn + 2$ or $O(n)$ messages to be exchanged, where $k \ll n$ is a constant that represents the maximum number of source agents that a feedback provider may trust to preserve its privacy. In experiment 4 (Section 4.8), we observe that k can be set as low as 2, while preserving the privacy of a high majority of agents.

The k-shares protocol is similar to the protocol in [105, section 5.2] in several aspects. Both protocols use secret sharing and compute reputation in a decentralized additive manner. However, there are also some key differences between the two protocols.

The protocol in [105] depends on a witness selection scheme and requires each feedback provider to exchange messages with all other $n-1$ source agents in the protocol. The message complexity of the protocol is therefore a high $O(N) + O(n^2)$. In contrast, the k-Shares protocol requires an exchange of only $O(n)$ number of messages. This is achieved by placing a constant limit ($k \ll n$) on the number of source agents that each feedback provider exchanges messages with, and by relying on trust awareness instead of a witness selection scheme. As we observe in Section 4.8.5, the privacy of a high majority of agents can be ensured with k as small as 2. Moreover, increasing k to values approaching $n-1$ has no significant advantage.

The k-Shares protocol may also be extended such that agents are allowed to abstain when they don't find trustworthy agents. In that case, an agent would generate two shares whose sum equals zero. One of the shares would be sent to a random source agent and the other to the querying agent with any shares received added to it. The abstaining agent would inform the querying agent that it has abstained. We note that allowing agents to abstain from providing feedback has an effect on the correctness of the reputation values computed by the protocol. The sum of the feedback values would change, however, it is possible that the reputation computed as mean remains unchanged or close to the actual value. In Section 4.5.1, we define metrics to measure the correctness of the reputation values computed when agents are allowed to abstain.

In the k-Shares protocol, an agent is able to quantify the privacy guarantee as subjective probability before it relies on trustworthy agents to preserve its privacy. This enables the agent (in the extended version) to abstain from providing its feedback value if the privacy guarantee is insufficient.

Allowing agents to abstain implies that their privacy can be preserved up to

100%. Perfect privacy cannot be achieved by the protocols presented in [105]. This topic is discussed in further detail in Section 4.5.1.

4.7 Determining the Suitability of the k-Shares Protocol for a Given Web of Trust

4.7.1 Instances of Source Agents whose Privacy will be Preserved

In the protocol Semi-Honest-k-Shares, the following assumption must hold for an agent $\boldsymbol{a}$'s privacy to be preserved: $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$ is low. That is, the probability that all agents whom agent $\boldsymbol{a}$ trusts are dishonest must be low.

In the following section we develop an algorithm that determines the percentage of instances of source agents in a given web of trust for whom this assumption indeed holds true. The k-Shares protocol is suitable for the web of trust if this percentage is sufficiently high.

Global Algorithm

We describe a general algorithm that helps determine whether the protocol Semi-Honest-k-Shares is suitable for a given web of trust. Consider a scenario in which the reputation of every potential target agent in the web of trust is queried once. This scenario instantiates all possible instances of source agents in the given web of trust. The algorithm operates on any web of trust (Definition 7) and returns the percentage of instances of source agents whose privacy will be preserved in such a scenario.

The algorithm takes the following four variables as input: G_ψ, MIN, LOW, and k. G_ψ is the web of trust, with context ψ. The algorithm considers an agent as a target agent if it has at least MIN source agents. LOW is the value of $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$, below or equal to which the privacy of an agent $\boldsymbol{a}$ is considered to be preserved. k is the limit on the number of fellow source agents that an agent $\boldsymbol{a}$ can rely on for its privacy to be preserved.

Let a set $O_{u,\psi} = \{v \mid \text{trust of } u \text{ in } v \text{ exists in context } \psi\}$. In other words, $O_{u,\psi}$ is the set of all agents for whom agent u is a source agent in the context ψ. $O_{u,\psi}$ may be noted as O_u when the context ψ is clear.

The algorithm is specified in Figure 4.9.

Local Algorithm

The algorithm specified in Figure 4.9 requires global knowledge of the web of trust. However, this may not be possible, especially since the privacy of agents is the main concern.

```
PRIVACY-PRESERVED-PC-GLOBAL(G_ψ, MIN, LOW, k)
 1  preserved-instances ← 0
 2  total-instances ← 0
    ▷ for each agent who has assigned feedback to at least one other agent
 3  for each agent a ∈ A, where |O_a − {a}| > 0
 4        do ▷ for each agent who has been assigned at least MIN feedback values
 5           for each agent t ∈ O_a − {a}, where |S_t| ≥ MIN
 6                 do total-instances ← total-instances +1
                       ▷ determine the set of candidate trustworthy agents
 7                    I ← (O_a − {t}) ∩ (S_t − {a})
                       ▷ create a vector of feedback values assigned to agents in I
 8                    V⃗ ← ⟨l_{as_1} ... l_{as_|I|}⟩, where I = {s_1 ... s_|I|}
 9                    sort V⃗ in descending order
10                    privacy-risk ← 1
11                    i ← 1
12                    privacy-risk ← privacy-risk ×(1 − V⃗[i])
13                    while (privacy-risk > LOW) and (i < k) and (i < |I|)
14                          do i ← i + 1
15                             privacy-risk ← privacy-risk ×(1 − V⃗[i])
16                    if privacy-risk ≤ LOW
17                      then preserved-instances ← preserved-instances +1
18  if total-instances ≠ 0
19     then return (preserved-instances / total-instances)%
20     else  return 0%
```

Figure 4.9: Percentage of Instances whose Privacy will be Preserved (Global).

Consider an alternate scenario in which the reputation of every potential target agent in the set O_a of an agent $\boldsymbol{a}$ is queried once. This scenario instantiates all possible instances of agent $\boldsymbol{a}$ as a source agent. We describe a general algorithm that returns the percentage of instances of agent $\boldsymbol{a}$ as a source agent whose privacy will be preserved in such a scenario. The algorithm operates only on the local knowledge of agent $\boldsymbol{a}$ and public knowledge.

If an agent finds that its percentage is too low, it knows that it can improve its percentage by building trust relationships with the agents in the set $\bigcup_{\forall t \in O_a} S_t$.

The algorithm takes as input the following three variables described previously: MIN, LOW, and k. Moreover, access to the local knowledge of $\boldsymbol{a}$, that is O_a and l_{ab}, for each $b \in O_a$, is assumed. S_t is public knowledge for each $t \in O_a$.

The algorithm is presented in Figure 4.10.

PRIVACY-PRESERVED-PC-LOCAL(MIN, LOW, k)

```
 1  preserved-instances ← 0
 2  total-instances ← 0
    ▷ for each agent who has been assigned at least MIN feedback values
 3  for each agent t ∈ O_a − {a}, where |S_t| ≥ MIN
 4        do total-instances ← total-instances +1
 5           ▷ determine the set of candidate trustworthy agents
 6           I ← (O_a − {t}) ∩ (S_t − {a})
             ▷ create a vector of feedback values assigned to agents in I
 7           V⃗ ← ⟨l_{as_1} ... l_{as_|I|}⟩, where I = {s_1 ... s_|I|}
 8           sort V⃗ in descending order
 9           privacy-risk ← 1
10           i ← 1
11           privacy-risk ← privacy-risk ×(1 − V⃗[i])
12           while (privacy-risk > LOW) and (i < k) and (i < |I|)
13                 do i ← i + 1
14                    privacy-risk ← privacy-risk ×(1 − V⃗[i])
15           if privacy-risk ≤ LOW
16              then preserved-instances ← preserved-instances +1
17  if total-instances ≠ 0
18     then return (preserved-instances / total-instances)%
19     else return 0%
```

Figure 4.10: Percentage of Instances whose Privacy will be Preserved (Local).

4.7.2 Convergence of k and the Percentage of Instances whose Privacy will be Preserved

We develop an algorithm (Figure 4.11) that takes any web of trust G_ψ and prints the values of k and the percentage of instances whose privacy will be preserved at their points of convergence. This algorithm helps determine the ideal value of k for a given web of trust. The local version of the algorithm is given in Figure 4.12.

The function *max-incoming(G)* takes a graph G and gives the maximum number of incoming edges for any vertex in the graph. In other words, the function returns the maximum number of source agents for any agent in the graph.

```
PRIVACY-PC-CONVERGENCE-GLOBAL(G_ψ, MIN, LOW, THRESHOLD)
 1  pc-convergence ← 0
 2  k-convergence ← 0
 3  k ← 1
 4  pc ← 0
    ▷ until the maximum possible value of k
 5  while k ≤ max-incoming(G_ψ)
 6        do pc-prev ← pc
 7           pc ← PRIVACY-PRESERVED-PC-GLOBAL(G_ψ, MIN, LOW, k)
 8           if pc − pc-prev ≥ THRESHOLD
 9              then pc-convergence ← pc
10                   k-convergence ← k
11           k ← k + 1
12 print pc-convergence, k-convergence
```

Figure 4.11: Convergence (Global).

4.8 Experiments

4.8.1 The Dataset: Advogato.org

We use the real web of trust of Advogato.org [87] as the dataset for our experiments. The members of Advogato rate each other in the context of being active and responsible members of the open source software developer community. The choice of feedback values are *master*, *journeyer*, *apprentice*, and *observer*, with *master* being the highest level in that order. The result of these ratings is a rich web of trust, which comprises of $13,904$ users and $57,114$ trust ratings (November 20, 2009). The distribution of ratings is as follows: *master*: 31.7%, *journeyer*: 40.3%, *apprentice*: 18.7%, and *observer*: 9.3%.

```
PRIVACY-PC-CONVERGENCE-LOCAL(MIN, LOW, THRESHOLD, MAX-K)
 1  pc-convergence ← 0
 2  k-convergence ← 0
 3  k ← 1
 4  pc ← 0
    ▷ until the maximum possible value of k
 5  while k ≤ MAX-K
 6        do pc-prev ← pc
 7           pc ← PRIVACY-PRESERVED-PC-LOCAL(MIN, LOW, k)
 8           if pc − pc-prev ≥ THRESHOLD
 9              then pc-convergence ← pc
10                   k-convergence ← k
11           k ← k + 1
12  print pc-convergence, k-convergence
```

Figure 4.12: Convergence (Local).

The members of Advogato are expected to not post spam, not attack the Advogato trust metric, etc. Thus we consider that on Advogato, the context "be a responsible member of the open source software developer community", comprises of the context "be honest". Since we quantify trust as probability, we substitute the four feedback values of Advogato as follows: $master = 0.99$, $journeyer = 0.70$, $apprentice = 0.40$, and $observer = 0.10$. These substitutions are made heuristically based on our observation of the significance of these feedback values on Advogato.org.

For the experiments, we define the lowest acceptable probability that privacy will be preserved as 0.90. This means that a set of two trustworthy agents must include either one *master* rated agent or two *journeyer* rated agents for this threshold to be satisfied.

4.8.2 Experiment 1 – Semi-Honest-Round-Trip

Objective: In the protocol Semi-Honest-Round-Trip, the following assumption must hold for an agent $\boldsymbol{a}$'s privacy to be preserved: $P(perform(a, a_{(f,out)}, \rho) = false) \times P(perform(a, a_{(b,out)}, \rho) = false)$ is low. That is, the probability that the successors of agent $\boldsymbol{a}$ in the *forwards* trip and the *backwards* trip are both dishonest must be low. **We would like to know the percentage of instances of source agents for whom this assumption holds true.**

Algorithm: A randomly selected querying agent queries the reputation of every other agent who has at least min source agents. Over the course of all queries, we observe the probability $P(perform(a, a_{(f,out)}, \rho) = false) \times P(perform(a, a_{(b,out)}, \rho) = false)$, for each source agent $\boldsymbol{a}$. The experiment is run for each value of min from 5 to 50 with intervals of 5.

Results: For $min = 5$, we observe that the probability that privacy will be preserved is sufficient (≥ 0.90) for 38.3% of instances of source agents. Figure 4.13 depicts the percentage of instances of source agents whose privacy is preserved as we vary the minimum number of source agents. We note that the increase in the percentage is not very substantial as we move *min* from 5 to 50. The percentage stays around 40%, which is clearly quite low. This implies that about 60% of instances of source agents are not assured that their privacy will be preserved. These agents must abstain from providing their feedback in the extended version of the protocol. However, as we observe in the next experiment, even with around 40% of agents contributing, the disparity is very low for a high majority of the reputation values computed.

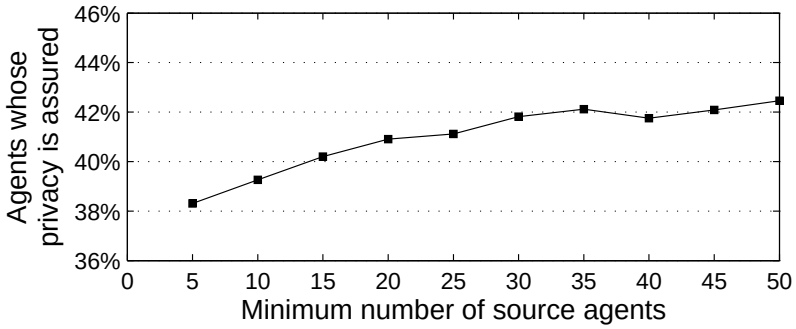

Figure 4.13: Semi-Honest-Round-Trip – Percentage of agents whose privacy is assured

4.8.3 Experiment 2 – Semi-Honest-Round-Trip-Abstain

Objective: We would like to know the resolution (Definition 24) and the disparity (Definition 25) of the reputation values computed by the protocol Semi-Honest-Round-Trip-Abstain.

Algorithm: A randomly selected querying agent queries the reputation of every other agent who has at least *min* source agents. Over the course of all queries, we observe the resolution and disparity of the reputation values computed by the protocol. The experiment is run for each value of *min* from 5 to 25 with intervals of 5.

Results: For $min = 25$, we observe that the resolution lies between 30% and 50% for 56% of the reputation values. The resolution lies between 20% and 60% for 89% of the reputation values. Clearly, the resolution is quite low most of the time. However, we also observe that for more than 95% of instances, the disparity is less than 0.10, and for 99.6% of the instances, the disparity is less than 0.2. Thus even though the resolution is low, a high majority of the reputation values computed are quite accurate. We hypothesize that the reason

for this accuracy is that the agents who are able to contribute represent a good sample of the total population of the source agents in the protocol. Figure 4.14 depicts the change in the percentage of reputation values with $disparity < 0.1$ and $disparity < 0.2$ as min varies from 5 to 25.

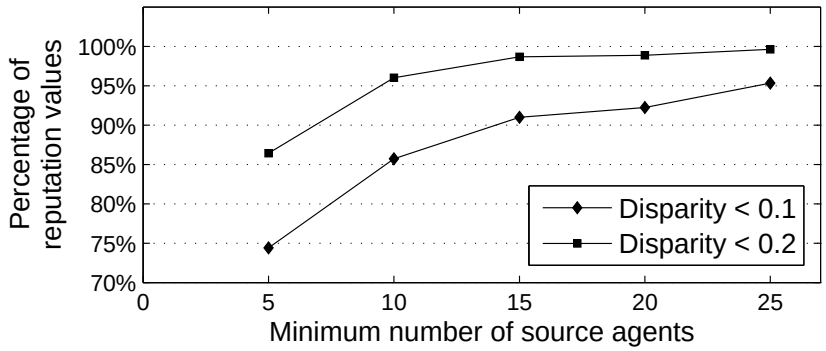

Figure 4.14: Semi-Honest-Round-Trip-Abstain – Percentage of reputation values with $disparity < 0.1$ and $disparity < 0.2$

4.8.4 Experiment 3 – Semi-Honest-k-Shares

Objective: In the protocol Semi-Honest-k-Shares, the following assumption must hold for an agent $\boldsymbol{a}$'s privacy to be preserved: $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$ is low. That is, the probability that the agents to whom agent $\boldsymbol{a}$ sends shares, are all dishonest must be low. **We would like to know the percentage of instances of source agents for whom this assumption holds true.**

Algorithm: A randomly selected querying agent queries the reputation of every other agent who has at least min source agents. Over the course of all queries, we observe the probability $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$, for each source agent $\boldsymbol{a}$. The experiment is run for each value of min in $\{1, 2, 3, 4, 5, 10, 15, 20, 25, 50, 100, 500\}$, with $k = 2$.

Results: For $min = 25$, we observe that the assumption holds for 81.7% of instances of source agents. Figure 4.15 depicts the percentage of instances of source agents whose privacy is preserved as we vary the minimum number of source agents, with $k = 2$. We note that the increase in the percentage is significant from $min = 5$ to $min = 100$. This is due to the greater choice of trustworthy agents available for each agent when the protocol has more source agents. However, even at $min = 5$, the percentage is 72.5%, which is almost twice as high as in the Round-Trip protocol. The remaining 10% to 30% of the source agents will have to abstain. However, as observed for the Round-Trip protocol, even a percentage of around 60% agents abstaining leads to quite accurate reputation values.

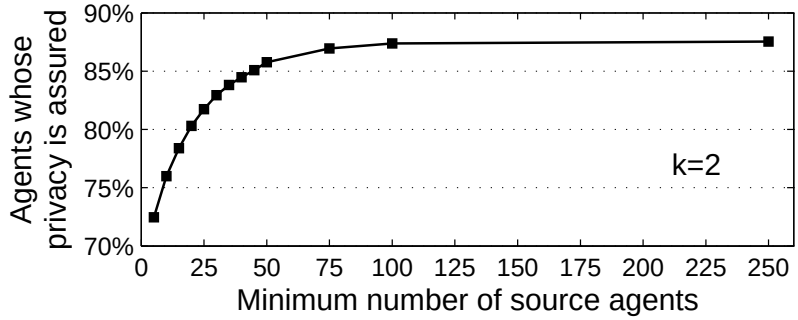

Figure 4.15: Semi-Honest-k-Shares – Percentage of agents whose privacy is assured

4.8.5 Experiment 4 – Semi-Honest-k-Shares

Objective: We would like to know the effect of increasing k on the percentage of instances of source agents whose privacy is preserved in the protocol Semi-Honest-k-Shares.

Algorithm: A randomly selected querying agent queries the reputation of every other agent who has at least min source agents. We vary k and observe the percentage of instances of source agents whose privacy is preserved. The set of experiments is first run with $min = 5$, and then with $min = 50$.

Results: For $min = 50$, and $k = 1$, we observe that the percentage is 75.4%, and at $k = 2$, the percentage is 85.8% (Figure 4.16). The jump is due to the possibility with $k = 2$ to rely on two *journeyer* agents. With $k = 1$, the only possibility is to rely on one *master* agent. However, increasing k over 2, even up to 500, does not result in a significant advantage. Thus, in this dataset, privacy can be preserved for a high percentage of source agents with k as small as 2. This results in a very efficient protocol. This is in contrast to the protocol presented in [105], which requires each agent to send shares to $n - 1$ agents, resulting in $O(N) + O(n^2)$ message complexity.

4.9 Discussion

Table 4.7 provides a comparison of our reputation protocols with the other systems in the literature.

Our reputation protocols, Semi-Honest-Round-Trip and Semi-Honest-k-Shares, are decentralized and provide security under the full semi-honest model. The protocols do not rely on TTPs or specialized networks. The number of messages required by each protocol is $O(n)$, where n is the number of source agents in the protocol.

In contrast, the systems discussed in the literature for the semi-honest model, either rely on TTPs or specialized networks, or require a high communication

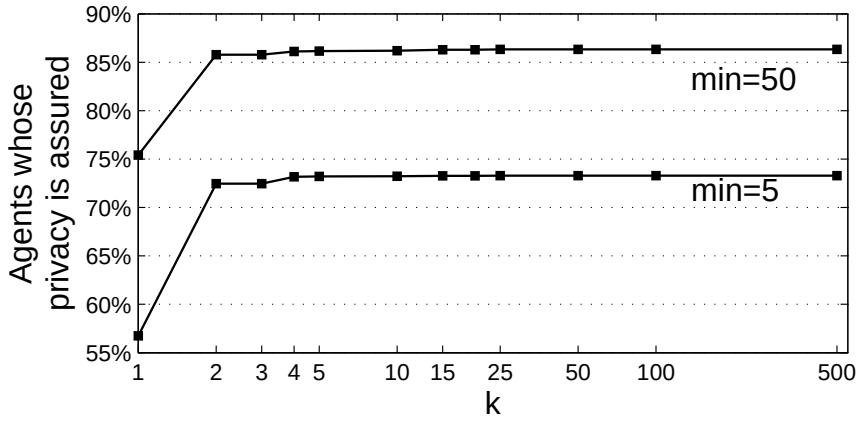

Figure 4.16: Semi-Honest-k-Shares – Effect of increasing k on the percentage of agents whose privacy is assured

complexity. The Secure Sum protocol [31] provides security under the semi-honest model only with the added restriction that agents do not collude. The scheme based on WSS-1 by Pavlov et al. [105] and the scheme 3 by Gudes et al. [61] require at least $O(n^2)$ messages. The first two schemes by Gudes et al. [61] and the reputation system by Nin et al. [101] are efficient, however, they either rely on TTPs or specialized networks.

It is assumed in [105] that it is not possible to achieve perfect privacy in decentralized additive reputation systems. However, the extended versions of our protocols provide up to perfect privacy by allowing feedback providers to quantify their privacy guarantee and to abstain if it is found to be insufficient.

Table 4.7: Protocols for the Semi-Honest Adversarial Model – Comparison.

System / Protocol	**Architecture**	**Target Environment**	**Key Security Mechanisms**	**Privacy Guarantee**	**Complexity (Messages)**
Semi-Honest-Round-Trip	D	Distributed environments	Secure multi-party computation, Trust awareness, Data perturbation	If $h \geq \Upsilon$ and $P(perform(a, a_{(f,out)}, \rho) = false) \times P(perform(a, a_{(b,out)}, \rho) = false)$ is low for each $a \in S_t$	$O(n)$
Semi-Honest-k-Shares	D	Distributed environments	Secure multi-party computation, Trust awareness, Secret sharing	If $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$ is low for each $a \in S_t$	$O(n)$
Clifton et al. [31] – Secure Sum	D	Distributed environments	Secure multi-party computation	Probability: $\frac{1}{m+1}$, only if nodes don't collude	$O(n)$, where n = number of sites
Pavlov et al. [105] – WSS-1	D	Distributed environments	Secure multi-party computation, secret sharing	$(1 - \frac{1}{n})(\frac{N-b-1}{N-1})$	$O(N) + O(n^2)$, where N = no. of potential witnesses, and n = no. of selected witnesses
Gudes et al. [61] – Scheme 1	D	Distributed environments	TTP, Public-key cryptography	Random guess across $\lvert TrustSet_x(A)\rvert$	$O(n)$, where $n = \lvert TrustSet_x(A)\rvert$
Gudes et al. [61] – Scheme 2	D	Distributed environments	TTP, Public-key cryptography, Secure product	Random guess across $\lvert TrustSet_x(A)\rvert$	$O(n)$, where $n = \lvert TrustSet_x(A)\rvert$
Gudes et al. [61] – Scheme 3	D	Distributed environments	Secure multi-party computation	A does not learn more information about $DTE(B, x)$, where $B \in TrustSet_x(A)$	$O(n^2)$, where $n = \lvert TrustSet_x(A)\rvert$
Nin et al. [101]	D	Private collaborative networks	ElGamal encryption scheme	If the underlying encryption scheme is secure	$O(1)$

Chapter 5

Reputation Protocols for the Malicious Adversarial Models

In this chapter, we develop privacy preserving reputation protocols that are secure under the stricter malicious adversarial models.

We begin the chapter with the problem definition. We then provide an ideal privacy preserving reputation protocol in the disruptive malicious model that relies on a trusted third party for security.

The tools and techniques that we use for the construction of the real privacy preserving reputation protocol in the disruptive malicious model include additive homomorphic cryptosystems, randomized encryption, and zero-knowledge proofs. These tools and techniques are discussed in Section 5.3.

In sections 5.4 and 5.5, we present the real privacy preserving reputation protocols that are secure under the non-disruptive and the disruptive malicious models respectively. We show that the real protocol for the disruptive malicious model is secure by demonstrating that it is able to emulate the ideal protocol for this model.

We conclude the chapter with a comparison of our new protocol for the disruptive malicious model with the existing protocols.

5.1 Problem Definition

Definition 26. ***Problem Definition (Disruptive Malicious).*** *Let $S_{t,\psi} = \{a_1 \dots a_n\}$ be the set of all source agents of agent t in the context of action ψ. Find a reputation protocol Π, which takes private input $l_{at} \equiv P(perform(a, t, \psi) = true)$ from each agent $a \in S_t$, and outputs the reputation $r_{t,\psi}$ of the target agent t to a querying agent q. Reputation is computed as $rep_{\oplus}$ (Equation 3.2). Agents q, t, and $m < n$ of the source agents (given as set $\mathbb{M}$) are considered*

to be dishonest, however, q wishes to learn the correct output. The reputation protocol Π *is required to be secure under the disruptive malicious adversarial model. If computing* $r_{t,\psi}$ *is not possible due to the disruptive actions of certain agents, then instead of the reputation, the protocol outputs the identity of those agents to the querying agent* q*.*

5.2 An Ideal Reputation Protocol

We first present an ideal privacy preserving reputation protocol in the disruptive malicious model. The subsequent real reputation protocol (Section 5.5) will attempt to emulate the privacy preservation property of this protocol.

5.2.1 Protocol Specification

The protocol is specified in Figure 5.1.

5.2.2 Discussion

The TTP_{S_t} is 100% trustworthy (Definition 18). It performs all computations correctly and preserves the privacy of all agents.

The protocol is correct since it outputs the correct reputation ($r_{t,\psi} = (\sum_{a \in S_t} l_{at})/n$, Equation 3.2) of the target agent to the querying agent. In the scenario where the protocol is unable to compute the correct reputation due to disruptive actions of source agents, the protocol outputs the identities of those agents to the querying agent. The disruptive agents are identified easily since all communication goes through the TTP_{S_t} and it is able to verify the correctness of the submitted feedback values. Communication takes place over authenticated point-to-point communication channels that are resistant to tampering, which ensures the integrity of the submitted feedback values.

The privacy of honest source agents is preserved since over the course of the protocol, they submit their private feedback only to the TTP_{S_t} over authenticated point-to-point communication channels that are resistant to wire-tapping.

The querying agent may re-initiate the protocol until no more disruptive agents are present and it receives the reputation as output.

Please note that the "ideal" protocol is ideal only in terms of preserving the privacy of agents. The protocol does not attempt to address issues other than the lack of privacy, such as the sybil attack, slandering, self-promotion, etc. (these issues are discussed in Section 2.2.3).

5.3 Tools and Techniques

In this section we discuss several tools and techniques that we use for the construction of the real privacy preserving reputation protocol under the disruptive malicious adversarial model (Section 5.5).

Protocol: Disruptive-Malicious-Ideal
Participants: Agents: q, t, $S_t \equiv S_{t,\psi} = \{a_1 \dots a_n\}$, TTP_{S_t}. Agents q, t, and a subset of $S_{t,\psi}$ of size $m < n$ are considered to be dishonest, however, q wishes to learn the correct output. $n \geq 3$.
Input: Each source agent $\boldsymbol{a}$ has a private input $l_{at} \equiv P(perform(a, t, \psi) = true)$.
Output: Agent q learns $r_{t,\psi}$, the reputation of agent t in the context ψ, or agent q learns the identity of the agents who disrupt the protocol.
Setup: The TTP_{S_t} maintains S_t. When an agent $\boldsymbol{a}$ assigns feedback to agent t, agent $\boldsymbol{a}$ reports the event to the TTP_{S_t}, which updates S_t accordingly. All participants know the identity of the TTP_{S_t}. All communication takes place over authenticated point-to-point channels that are resistant to wire-tapping and tampering.
Steps:
1. The querying agent q sends a request to the TTP_{S_t} for the reputation $r_{t,\psi}$ of agent t.
2. The TTP_{S_t} requests each agent $a \in S_t$ for feedback l_{at}.
3. Each agent $a \in S_t$ sends l_{at} to the TTP_{S_t}.
4. After receiving l_{at} from each agent $a \in S_t$, the TTP_{S_t} computes the reputation of agent t as $r_{t,\psi} = (\sum_{a \in S_t} l_{at})/n$.
5. The TTP_{S_t} sends $r_{t,\psi}$ to q.
6. If there are source agents who disrupt the protocol by either not sending the feedback to the TTP_{S_t} within a predetermined amount of time, or by providing out of range feedback, then the TTP_{S_t} is unable to compute the correct $r_{t,\psi}$. In this scenario, the TTP_{S_t} sends the identities of these disruptive agents to q. Moreover, the TTP_{S_t} removes these agents from S_t for future queries by q for $r_{t,\psi}$.

Figure 5.1: Protocol: Disruptive-Malicious-Ideal

5.3.1 Additive Homomorphic Cryptosystems

Let $E_a(.)$ denote the encryption function with the public key PK_a of agent $\boldsymbol{a}$ in an asymmetric cryptosystem $\mathcal{C}$. The cryptosystem $\mathcal{C}$ is said to be additive homomorphic if we can compute $E_a(x+y)$, given only $E_a(x)$, $E_a(y)$, and PK_a. In other words, a cryptosystem is additive homomorphic if we can compute the encryption of the sum of two plaintexts, given only their ciphertexts and the encrypting public key. As an example, let's consider two integers, 3 and 4. A cryptosystem $\mathcal{C}$ is additive homomorphic if given only $E_a(3)$, $E_a(4)$, and PK_a, we are able to obtain $E_a(3+4) = E_a(7)$.

The Paillier cryptosystem [104] (described in Section 5.3.4) is a well-known additive homomorphic cryptosystem. Other examples include the Okamoto-Uchiyama cryptosystem [102] (a historical antecedent of Paillier), and the

Damgård-Jurik cryptosystem [34] (a generalization of Paillier).

5.3.2 Randomized Encryption

A randomized encryption function [115] generates the ciphertext for a given plaintext and key in a non-deterministic manner. This implies that there may exist several possible ciphertexts for the same plaintext and key. However, each ciphertext generated by a randomized encryption function corresponds to only one plaintext. As an example, consider $c_1 = E_a(3)$ and $c_2 = E_a(3)$. With randomized encryption, it is possible that $c_1 \neq c_2$. However, both c_1 and c_2 will decrypt to the same plaintext, that is, 3.

The advantage of randomized encryption is that an attacker cannot distinguish between the encryptions of different plaintexts even if the plaintexts and the key are known. For example, consider an attacker who is given two integers, 3 and 4, their encryptions, $E_a(3)$ and $E_a(4)$, and the encrypting public key PK_a. Randomized encryption implies that the attacker is unable to draw correspondence between the ciphertexts $E_a(3)$, $E_a(4)$, and the integers 3, 4.

Cryptosystems that do not support randomized encryption (for example, RSA [114] without padding), always generate the same ciphertext for a given pair of plaintext and encryption key. Such cryptosystems are not suitable when the plaintext space is small (for example, a plaintext space such as $\{1, 2, 3, 4, 5\}$). The adversary can easily compute the ciphertext for each plaintext. Then given any ciphertext, it can deduce the corresponding plaintext.

5.3.3 Semantic Security

An asymmetric cryptosystem is said to be semantically secure if given a ciphertext and the encrypting public key, a computationally bounded adversary is unable to learn any significant information about the plaintext.

A semantically secure cryptosystem is secure against the Chosen-Plaintext Attack (CPA). The chosen-plaintext attack is an attack model for cryptanalysis in which the attacker is able to choose any plaintext and observe its corresponding ciphertext.

A semantically secure cryptosystem is not secure against the Chosen-Ciphertext Attack (CCA). This attack-model assumes that the attacker is able to choose any ciphertext and obtain its corresponding plaintext without the knowledge of the secret key. The CPA and CCA are discussed in detail in [56, Chapter 5].

Semantic security is sufficient when the chosen-ciphertext attack can be prevented. This can be achieved by avoiding situations in which the adversary is able to request the decryption of any chosen ciphertext.

Examples of semantically secure cryptosystems include Goldwasser-Micali [57], ElGamal [46], and Paillier [104] (described in Section 5.3.4). A cryptosystem that is not semantically secure is the RSA cryptosystem [114]. It is thus vulnerable even to the chosen-plaintext attack. However, practical implementa-

tions of RSA utilize padding (augmenting the plaintext with random data) to provide security under this attack model.

5.3.4 The Paillier Cryptosystem

The Paillier cryptosystem is an additive homomorphic cryptosystem. It provides randomized encryption and is semantically secure. The scheme is described in Figure 5.2, where $\mathbb{k}$, $\mathfrak{m}$, and c, are the security parameter, the plaintext, and the ciphertext respectively.

One of the building blocks of our protocol (Section 5.5) for the disruptive malicious model is an additive homomorphic cryptosystem with the characteristics listed in Section 5.5.1. The Paillier cryptosystem conforms to these requirements and therefore we use it as the additive homomorphic cryptosystem for our protocol.

Key Generation

1. Select $\mathbb{k}$, the length in bits of an RSA modulus $\mathfrak{n}$
2. Select two random and distinct primes p and q of length $\mathbb{k}/2$
3. Compute $\mathfrak{n} = p \cdot q$
4. Compute $\lambda = lcm(p-1, q-1) = \frac{(p-1)(q-1)}{gcd(p-1,q-1)}$
5. Select a random integer $g \in \mathbb{Z}^*_{\mathfrak{n}^2}$
6. Compute $\mu = (L(g^\lambda \ mod \ \mathfrak{n}^2))^{-1} \ mod \ \mathfrak{n}$, where $L(u) = (u-1)/\mathfrak{n}$
7. Public key: $(\mathfrak{n}, g)$, private key: (λ, μ)

Encryption

1. Select a random integer $\mathfrak{r}$ in $\mathbb{Z}^*_{\mathfrak{n}}$
2. Compute ciphertext $c = g^{\mathfrak{m}} \cdot \mathfrak{r}^{\mathfrak{n}} \ mod \ \mathfrak{n}^2$

Decryption

1. Compute plaintext $\mathfrak{m} = L(c^\lambda \ mod \ \mathfrak{n}^2) \cdot \mu \ mod \ \mathfrak{n}$, where $L(u) = (u-1)/\mathfrak{n}$

Additive Homomorphic Property

Given two integers $\mathfrak{m}_1$ and $\mathfrak{m}_2$, and their respective ciphertexts c_1 and c_2, the following holds (D is the decryption function):

$\mathfrak{m}_1 + \mathfrak{m}_2 \ mod \ \mathfrak{n} = D(c_1 \cdot c_2 \ mod \ \mathfrak{n}^2)$

Figure 5.2: The Paillier Cryptosystem

Our implementation of an interactive demo of the Paillier cryptosystem is available online [66].

5.3.5 Zero-Knowledge Proofs

A zero-knowledge proof [58] is an interactive proof that allows a prover to convince a verifier that a statement is true without revealing any information other than the fact that the statement is valid.

As an example, consider a prover who knows an RSA modulus $\mathtt{n}$ and its two large prime factors p and q. A verifier knows only $\mathtt{n}$. Factoring $\mathtt{n}$ is considered intractable therefore the verifier cannot learn p and q. An interactive proof would be zero-knowledge if it allows the prover to convince the verifier that he knows the factors of $\mathtt{n}$ without revealing any information about p and q.

A standard interactive zero-knowledge proof comprises of three moves, that is, three messages exchanged between the prover and the verifier. In the first move, the prover sends a cryptographic commitment to the verifier. In the second move, the verifier sends a random challenge to the prover to test the commitment. The third move is the prover's response to the random challenge of the verifier.

An interactive zero-knowledge proof can be converted to a non-interactive zero-knowledge proof using the Fiat-Shamir heuristic [49]. A non-interactive zero-knowledge proof comprises of only one move, that is, one message sent by the prover to the verifier. An interactive zero-knowledge proof requires three moves because the verifier must provide a random challenge to the prover. In the non-interactive version, this random challenge is replaced with a hash of the prover's commitment that the prover can generate itself. The prover generates the commitment, the challenge as the hash of the commitment, and the response to the challenge. It then sends everything in one move to the verifier. In turn, the verifier independently computes the hash and verifies the proof.

5.3.6 Zero-Knowledge Proof of Set Membership

In general, a zero-knowledge proof of set membership may be stated as follows:

Let $S = \{\mathtt{m}_1, \ldots, \mathtt{m}_p\}$ be a public set of p messages, and $E(\mathtt{m}_i)$ an encryption of $\mathtt{m}_i$ with a prover's public key, where i is secret. A zero-knowledge proof of set membership allows the prover to convince a verifier that $E(\mathtt{m}_i)$ encrypts a message in S.

In a non-interactive version of the zero-knowledge proof of set membership, we abstract the part of the proof generated by the prover as the function $setMembershipZKP(E(\mathtt{m}_i), S)$, abbreviated as $smzkp(E(\mathtt{m}_i), S)$.

The zero-knowledge proof of set membership is specified for the Paillier cryptosystem as follows:

Let $(\mathtt{n}, g)$ be a prover's public key, $S = \{\mathtt{m}_1, \ldots, \mathtt{m}_p\}$ be a public set of p messages, and $c = g^{\mathtt{m}_i} \cdot \mathtt{r}^{\mathtt{n}} \ mod \ \mathtt{n}^2$ an encryption of $\mathtt{m}_i$, where i is secret, and $\mathtt{r}$ is a random integer. A zero-knowledge proof of set membership allows the prover to convince a verifier that c encrypts a message in S.

An interactive zero-knowledge proof of set membership for the Paillier cryptosystem is described in [13]. In Figures 5.3 and 5.4, we present the interactive version and our non-interactive version of the proof respectively. We have ob-

tained the non-interactive version of the proof using the Fiat-Shamir heurisitic. To make the proof non-interactive, the challenge by the verifier in step 3 (move 2) has been replaced by a hash of the commitment.

The complexity (in terms of bandwidth used) of the proof described in [13] is $O(p)$, that is, linear with respect to the cardinality of the set S.

Protocol: Interactive-ZKP-Set-Membership
Participants: A prover and a verifier.
Input: Prover: $\mathfrak{n}$, g, $\mathfrak{m}_i$, p, $\mathfrak{r}$, $c = g^{\mathfrak{m}_i} \cdot \mathfrak{r}^{\mathfrak{n}} \ mod \ \mathfrak{n}^2$. Verifier: $\mathfrak{n}$, g, p, c.
Output: The verifier is convinced that c encrypts a message in S.
Setup: Public knowledge: A set $S = \{\mathfrak{m}_1, \ldots, \mathfrak{m}_p\}$, and the prover's public key $(\mathfrak{n}, g)$.
Steps:

Prover

1. Prover picks at random ρ in $\mathbb{Z}^*_{\mathfrak{n}}$
2. Prover randomly picks $p-1$ values e_j in $\mathbb{Z}_{\mathfrak{n}}$, where $j \neq i$
3. Prover randomly picks $p-1$ values v_j in $\mathbb{Z}^*_{\mathfrak{n}}$, where $j \neq i$
4. Prover computes $u_j = v_j^{\mathfrak{n}} \cdot (g^{\mathfrak{m}_j}/c)^{e_j} \ mod \ \mathfrak{n}^2$, where $j \neq i$, and $u_i = \rho^{\mathfrak{n}} \ mod \ \mathfrak{n}^2$
5. Move 1: Prover sends u_j, where j in $\{1 \ldots p\}$, to the verifier

Verifier

1. Verifier chooses a random challenge e in $[0, A[$
2. Move 2: Verifier sends e to the prover

Prover

1. Prover computes $e_i = e - \sum_{j \neq i} e_j \ mod \ \mathfrak{n}$
2. Prover computes $v_i = \rho \cdot \mathfrak{r}^{e_i} \cdot g^{(e - \sum_{j \neq i} e_j)/\mathfrak{n}} \ mod \ \mathfrak{n}$
3. Move 3: Prover sends v_j, e_j, where $j \in \{1 \ldots p\}$, to the verifier

Verifier

1. Verifier checks that $e = \sum_j e_j \ mod \ \mathfrak{n}$
2. Verifier checks that $v_j^{\mathfrak{n}} = u_j \cdot (c/g^{\mathfrak{m}_j})^{e_j} \ mod \ \mathfrak{n}^2$ for each $j \in \{1 \ldots p\}$

Figure 5.3: Protocol: Interactive Zero-Knowledge Proof of Set Membership [13]

5.3.7 Zero-Knowledge Proof of Plaintext Equality

In general, a zero-knowledge proof of plaintext equality for two ciphertexts may be stated as follows:

Let $E_1(\mathfrak{m})$ and $E_2(\mathfrak{m})$ be encryptions of a message $\mathfrak{m}$ with the public key of agents 1 and 2 respectively. A zero-knowledge proof of plaintext equality allows a prover to convince a verifier that $E_1(\mathfrak{m})$ and $E_2(\mathfrak{m})$ encrypt the same message.

Protocol: Non-Interactive-ZKP-Set-Membership
Participants: A prover and a verifier.
Input: Prover: $\mathfrak{n}$, g, $\mathfrak{m}_i$, p, $\mathfrak{r}$, $c = g^{\mathfrak{m}_i} \cdot \mathfrak{r}^{\mathfrak{n}} \ mod \ \mathfrak{n}^2$. Verifier: $\mathfrak{n}$, g, p, c.
Output: The verifier is convinced that c encrypts a message in S.
Setup: Public knowledge: A set $S = \{\mathfrak{m}_1, \ldots, \mathfrak{m}_p\}$, and the prover's public key $(\mathfrak{n}, g)$. $hash(x)$ is a cryptographic hash function secure against a computationally PPT bounded adversary.
Steps:

Prover

1. Prover picks at random ρ in $\mathbb{Z}^*_{\mathfrak{n}}$
2. Prover randomly picks $p-1$ values e_j in $\mathbb{Z}_{\mathfrak{n}}$, where $j \neq i$
3. Prover randomly picks $p-1$ values v_j in $\mathbb{Z}^*_{\mathfrak{n}}$, where $j \neq i$
4. Prover computes $u_j = v_j^{\mathfrak{n}} \cdot (g^{\mathfrak{m}_j}/c)^{e_j} \ mod \ \mathfrak{n}^2$, where $j \neq i$, and $u_i = \rho^{\mathfrak{n}} \ mod \ \mathfrak{n}^2$
5. Prover computes $e = hash(\langle u_1 \ldots u_p \rangle)$
6. Prover computes $e_i = e - \sum_{j \neq i} e_j \ mod \ \mathfrak{n}$
7. Prover computes $v_i = \rho \cdot \mathfrak{r}^{e_i} \cdot g^{(e - \sum_{j \neq i} e_j)/\mathfrak{n}} \ mod \ \mathfrak{n}$
8. Move 1: Prover sends u_j, v_j, e_j, where $j \in \{1 \ldots p\}$, to the verifier

Verifier

1. Verifier computes $e = hash(\langle u_1 \ldots u_p \rangle)$
2. Verifier checks that $e = \sum_j e_j \ mod \ \mathfrak{n}$
3. Verifier checks that $v_j^{\mathfrak{n}} = u_j \cdot (c/g^{\mathfrak{m}_j})^{e_j} \ mod \ \mathfrak{n}^2$ for each $j \in \{1 \ldots p\}$

Figure 5.4: Protocol: Non-Interactive Zero-Knowledge Proof of Set Membership

In a non-interactive version of the zero-knowledge proof of plaintext equality, we abstract the part of the proof generated by the prover as the function $plaintextEqualityZKP(E_u(\mathfrak{m}), E_v(\mathfrak{m}))$, abbreviated as $pezkp(E_u(\mathfrak{m}), E_v(\mathfrak{m}))$.

The zero-knowledge proof of plaintext equality is specified for the Paillier cryptosystem as follows:

Let $(\mathfrak{n}_1, g_1)$ and $(\mathfrak{n}_2, g_2)$ be the public keys of agents 1 and 2 respectively. Given two encryptions $c_1 = g_1^{\mathfrak{m}} \cdot \mathfrak{r}_1^{\mathfrak{n}_1} \ mod \ \mathfrak{n}_1^2$ and $c_2 = g_2^{\mathfrak{m}} \cdot \mathfrak{r}_2^{\mathfrak{n}_2} \ mod \ \mathfrak{n}_2^2$, a zero-knowledge proof of plaintext equality allows a prover to convince a verifier that c_1 and c_2 encrypt the same message.

An interactive zero-knowledge proof of plaintext equality for the Paillier cryptosystem is described in [13]. In Figure 5.5, we present our non-interactive version of the proof, which has been obtained using the Fiat-Shamir heurisitic.

For two ciphertexts, the complexity (in terms of bandwidth used) of the proof described in [13] is $O(1)$, that is, constant.

Protocol: Non-Interactive-ZKP-Plaintext-Equality
Participants: A prover and a verifier.
Input: Prover: $\mathfrak{n}_1$, g_1, $\mathfrak{n}_2$, g_2, $\mathfrak{m}$, $\mathfrak{r}_1$, $\mathfrak{r}_2$, $c_1 = g_1^{\mathfrak{m}} \cdot \mathfrak{r}_1^{\mathfrak{n}_1} \ mod \ \mathfrak{n}_1^2$, $c_2 = g_2^{\mathfrak{m}} \cdot \mathfrak{r}_2^{\mathfrak{n}_2} \ mod \ \mathfrak{n}_2^2$. Verifier: $\mathfrak{n}_1$, g_1, $\mathfrak{n}_2$, g_2, c_1, c_2.
Output: The verifier is convinced that c_1 and c_2 encrypt the same message.
Setup: Public knowledge: The public keys $(\mathfrak{n}_1, g_1)$ and $(\mathfrak{n}_2, g_2)$. $hash(x)$ is a cryptographic hash function secure against a computationally PPT bounded adversary.
Steps:

Prover

1. Prover picks at random ρ in $[0, 2^{\mathbb{k}}[$
2. Prover randomly picks $s_1 \in \mathbb{Z}^*_{\mathfrak{n}_1}$ and $s_2 \in \mathbb{Z}^*_{\mathfrak{n}_2}$
3. Prover computes $u_j = g_j^{\rho} \cdot s_j^{\mathfrak{n}_j} \ mod \ \mathfrak{n}_j^2$, for each $j \in \{1, 2\}$
4. Prover computes $e = hash(\langle u_1, u_2 \rangle)$
5. Prover computes $z = \rho + \mathfrak{m} \cdot e$
6. Prover computes $v_j = s_j \cdot \mathfrak{r}_j^e \ mod \ \mathfrak{n}_j$, for each $j \in \{1, 2\}$
7. Move 1: Prover sends z, u_1, u_2, v_1, v_2 to the verifier

Verifier

1. Verifier computes $e = hash(\langle u_1, u_2 \rangle)$
2. Verifier checks that $z \in [0, 2^{\mathbb{k}}[$
3. Verifier checks that $g_j^z \cdot v_j^{\mathfrak{n}_j} = u_j \cdot c_j^e \ mod \ \mathfrak{n}_j^2$ for each $j \in \{1, 2\}$

Figure 5.5: Protocol: Non-Interactive Zero-Knowledge Proof of Plaintext Equality

5.4 The Non-Disruptive Malicious Model

Malicious agents may 1) refuse to participate in the protocol, 2) prematurely abort the protocol, 3) selectively drop messages that they are supposed to send, 4) tamper with the communication channels, 5) wiretap the communication channels, and 6) provide incorrect information (for example, provide out of range values as their inputs).

We have defined a non-disruptive malicious adversary as a malicious adversary who executes the cited malicious actions only if they lead to the disclosure of the inputs of honest agents. Non-disruptive agents have a single objective: learn the inputs of honest agents. They do not disrupt the normal function of the protocol other than to achieve this objective.

The Semi-Honest-k-Shares protocol (Section 4.6) provides security under the semi-honest model. We analyze the possible malicious actions that agents can take in the Semi-Honest-k-Shares protocol under the non-disruptive malicious model.

Agent t. To gain unfair advantage, agent t could drop the agents from S_t whom he thinks might have rated him poorly.

All Agents. Agents could wiretap communication channels to learn the shares of an agent under attack.

We propose the following extensions to secure the k-Shares protocol under the non-disruptive malicious adversarial model.

5.4.1 Source Managers

The set S_a is no longer maintained by agent $\boldsymbol{a}$. Instead, the set S_a is maintained for agent $\boldsymbol{a}$ by two or more other agents in the system independently of each other. Those agents are called the *source managers* of agent $\boldsymbol{a}$. The idea of source managers is inspired by score managers in EigenTrust [80].

When a source agent assigns feedback to a target agent, it reports that event to each of the source managers of the target agent. The source managers add the source agent to the set S_t that they each maintain for the target agent t.

Agent q retrieves the set S_t from the source managers of agent t. It is possible that a number of the source managers are colluding with agent t and thus drop agents from S_t as desired by t. To counter this problem, an agent that needs the set S_t, retrieves it from all the source managers of agent t and then takes the union of all those sets to get the final S_t. Thus even if a single source manager is honest, the final set S_t would include all source agents of agent t.

To assign and locate source managers, a Distributed Hash Table (DHT), such as Chord [128], is used. An agent's source managers are located by hashing the unique ID of the agent. Chord requires $O(log\ N)$ messages for a lookup, where N is the total number of agents in the system.

It is important to note that a source manager is not a special agent in the system. Source manager is only a role that any regular agent can perform. Moreover, a source manager is not considered to be honest. A querying agent will receive the correct set S_t as long as at least one of all the source managers of the target agent t is honest. A DHT is used to assign source managers, therefore an agent has no influence over who serves as its source manager.

Let's consider that there is an even probability that any given source manager is either honest or dishonest. Then the probability that at least one of all $\mathtt{m}_a$ source managers of an agent $\boldsymbol{a}$ will be honest is $1 - \frac{1}{2^{\mathtt{m}_a}}$. This probability is 75% at $\mathtt{m}_a = 2$, 97% at $\mathtt{m}_a = 5$, and 99% at $\mathtt{m}_a = 7$.

A disadvantage of using source managers is the additional overhead on agents who serve as source managers.

5.4.2 Secure Communication

Wiretapping may be prevented by requiring all messages to be exchanged via communication channels that are resistant to wiretapping. This can be achieved through a protocol such as SSL (Secure Sockets Layer) or IPSec.

5.5 The k-Shares Reputation Protocol for the Disruptive Malicious Model

We have defined a disruptive malicious adversary as a malicious adversary who has the following objectives: 1) learn the inputs of honest agents, and 2) disrupt the protocol for honest agents. The reasons for disrupting the protocol may range from gaining illegitimate advantage over honest agents to completely denying the service of the protocol to honest agents.

5.5.1 A Suitable Cryptosystem

To construct a reputation protocol that is secure under the disruptive malicious adversarial model, we utilize an asymmetric cryptosystem that satisfies the following properties:

- Additive homomorphic cryptography
- Randomized encryption
- At minimum, semantic security
- Efficient zero-knowledge proofs of set membership and plaintext equality

Let $\mathcal{C}$ be any asymmetric cryptosystem such that it has the properties described above. Let $E_a(.)$ denote the encryption function with the public key PK_a of agent $\boldsymbol{a}$ in $\mathcal{C}$, and let $D_a(.)$ denote the decryption function with the secret key SK_a of agent $\boldsymbol{a}$ in $\mathcal{C}$. A realization of $\mathcal{C}$ is the Paillier cryptosystem, which we use for our protocol.

5.5.2 Protocol Outline

The important steps of the protocol are outlined below.

1. **Initiation.** The protocol is initiated by a querying agent q to determine the reputation $r_{t,\psi}$ of a target agent t. Agent q retrieves $S_t \equiv S_{t,\psi}$, the set of source agents of agent t in the context ψ. Agent q verifies S_t from the source managers of t. Agent q then sends S_t to each agent $a \in S_t$.

2. **Select Trustworthy Agents.** Each agent $a \in S_t$ selects up to k other agents in S_t. Let's refer to these agents selected by $\boldsymbol{a}$ as the set $U_a = \{u_{a,1} \ldots u_{a,k_a}\}$, where $1 \leq k_a \leq k$. Agent $\boldsymbol{a}$ selects these agents such that: $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$ is low. That is, the probability that all of the selected agents will collude to break agent $\boldsymbol{a}$'s privacy is low.

3. **Prepare Shares.** Agent $\boldsymbol{a}$ then prepares $k_a + 1$ shares of its secret feedback value l_{at}. The shares, given as: $x_{a,1} \ldots x_{a,k_a+1}$, are prepared in the following manner: The first k_a shares are random numbers uniformly

distributed over a large interval (for example, $[0, 2^{32} - 1]$). The last share is selected as follows: $x_{a,k_a+1} = (l_{at} - \sum_{i=1}^{k_a} x_{a,i})\ mod\ M$, where M is a publicly known modulus.

The preparation of the shares in this manner implies that: $(\sum_{i=1}^{k_a+1} x_{a,i})\ mod\ M = l_{at}$. That is, the sum of the shares $mod\ M$ is equal to the feedback value. The sum of the shares, $\sum_{i=1}^{k_a+1} x_{a,i}$, lies in $[(h_a \times M), (h_a \times M) + L]$, where $h_a = (\sum_{i=1}^{k_a+1} x_{a,i})\ div\ M$, and $l_{at} \in [0, L]$.

Since each of the $k_a + 1$ shares is a number uniformly distributed over a large interval, no information about the secret can be learnt unless all of the shares are known.

4. **Encrypt Shares.** Agent $\boldsymbol{a}$ then encrypts each of the $k_a + 1$ shares with its own public key to obtain: $E_a(x_{a,1}) \ldots E_a(x_{a,k_a+1})$. It also encrypts each share $x_{a,i}$ with the public key of agent $u_{a,i}$, for $i \in \{1 \ldots k_a\}$, to obtain: $E_{u_{a,1}}(x_{a,1}) \ldots E_{u_{a,k_a}}(x_{a,k_a})$.

   In a later step, each encrypted share $E_{u_{a,i}}(x_{a,i})$ will be delivered to the agent $u_{a,i}$.

5. **Generate Zero-Knowledge Proofs.** Agent $\boldsymbol{a}$ computes: $\beta_a = (E_a(x_{a,1}) \times \ldots \times E_a(x_{a,k_a+1}))\ mod\ \mathfrak{n}_a^2$, where $\mathfrak{n}_a$ is the RSA modulus in the public key of agent $\boldsymbol{a}$. The result of this product is the encrypted sum of agent $\boldsymbol{a}$'s shares, that is $\beta_a = E_a(\sum_{i=1}^{k_a+1} x_{a,i})$ (due to the additive homomorphic property).

   Agent $\boldsymbol{a}$ then generates one non-interactive set membership zero-knowledge proof: $smzkp(\beta_a, [(h_a \times M), (h_a \times M) + L])$. The proof proves to a verifier that the ciphertext β_a encrypts a value that lies in $[(h_a \times M), (h_a \times M) + L]$. In other words, the proof shows that the ciphertext contains a valid feedback value (considering $mod\ M$).

   Agent $\boldsymbol{a}$ also generates k_a non-interactive plaintext equality zero-knowledge proofs. Each proof $pezkp(E_a(x_{a,i}), E_{u_{a,i}}(x_{a,i}))$, where $i \in \{1 \ldots k_a\}$, proves to a verifier that the two ciphertexts, one encrypted with the public key of $\boldsymbol{a}$ and the other encrypted with the public key of $u_{a,i}$, contain the same plaintext.

   A verifier who verifies these zero-knowledge proofs will be convinced that agent $\boldsymbol{a}$ has prepared the shares such that they add up to a correct feedback value. Moreover, the verifier will be assured that the shares destined for $\boldsymbol{a}$'s trustworthy agents correspond to those correct shares.

6. **Send Encrypted Shares and Proofs.** Agent $\boldsymbol{a}$ sends all encrypted shares, that is, $E_a(x_{a,1}) \ldots E_a(x_{a,k_a+1})$ and $E_{u_{a,1}}(x_{a,1}) \ldots E_{u_{a,k_a}}(x_{a,k_a})$, as well as all zero-knowledge proofs, that is, $smzkp(\beta_a, [(h_a \times M), (h_a \times M) + L])$ and $pezkp(E_a(x_{a,i}), E_{u_{a,i}}(x_{a,i}))$, $i \in \{1 \ldots k_a\}$, to agent q.

7. **Verify the Proofs.** Agent q independently computes β_a and verifies the proofs received from each agent $\boldsymbol{a}$. Their verification confirms that agent $\boldsymbol{a}$

has prepared the shares correctly. Agent q receives and verifies the proofs of all source agents before proceeding to the next step.

8. **Relay the Encrypted Shares.** Agent q relays to each agent $\boldsymbol{a}$, the encrypted shares received for it from agents who considered it trustworthy. That is, each encrypted share $E_{u_{v,j}}(x_{v,j})$, prepared by an agent v for agent $u_{v,j}$, is relayed to agent $u_{v,j}$.

   Since the shares are relayed through q, any agent who drops a message would be easily identified. However, q does not learn any of the shares by relaying them since they are encrypted.

9. **Compute Sum of the Shares.** Each agent $\boldsymbol{a}$ receives the encrypted shares of the agents who considered it trustworthy. Agent $\boldsymbol{a}$ computes γ_a as the product of those encrypted shares along with the ciphertext of its own $k_a + 1$'th share x_{a,k_a+1}. Due to the additive homomorphic property, γ_a is an encryption of the sum of the plaintexts of those shares. Agent $\boldsymbol{a}$ decrypts γ_a to obtain the plaintext sum σ_a.

   Adding the $k_a + 1$'th share provides security in the case when $\boldsymbol{a}$ receives only one share. If there is no $k_a + 1$'th share to add, then q would learn the received share. Secrecy of the $k_a + 1$'th share itself is not critical to the security of the protocol.

10. **Encrypt the Sum.** Agent $\boldsymbol{a}$ then encrypts σ_a with agent q's public key to obtain $E_q(\sigma_a)$.

11. **Generate Zero-Knowledge Proof.** Agent $\boldsymbol{a}$ then generates a non-interactive plaintext equality zero-knowledge proof: $pezkp(\gamma_a, E_q(\sigma_a))$. The proof proves to a verifier that the two ciphertexts, one encrypted with the public key of $\boldsymbol{a}$ and the other encrypted with the public key of q, contain the same plaintext.

    Agent q, who can independently compute γ_a, can be convinced by this proof that $E_q(\sigma_a)$ contains the correct sum of the shares.

12. **Send Encrypted Sum and Proof.** Agent $\boldsymbol{a}$ sends the encrypted sum $E_q(\sigma_a)$ and the zero-knowledge proof $pezkp(\gamma_a, E_q(\sigma_a))$ to agent q.

13. **Verify the Proof.** Agent q independently computes γ_a and verifies the zero-knowledge proof received from each agent $\boldsymbol{a}$. Its verification confirms that the agent has computed the sum of the shares correctly. Agent q receives and verifies the proofs of all source agents before proceeding to the next step.

14. **Compute Reputation.** Agent q decrypts $E_q(\sigma_a)$ to obtain σ_a for each agent $a \in S_t$. Agent q then computes $r_{t,\psi} = ((\sum_{a \in S_t} \sigma_a)\ mod\ M)/n$.

5.5.3 Protocol Specification

The protocol is specified in Figures 5.6 through 5.9. Tables 5.1 and 5.2 describe the functions and the variables used in the protocol respectively.

The Paillier cryptosystem can only encrypt integers. Therefore, for the purpose of this protocol, we consider feedback values to be integers in the range $[0, L]$ (for example, $[0, 10]$). The reputation computed by the protocol can be normalized to the interval $[0, 1]$ by dividing the result by L.

Let M be a publicly known modulus, such that $M > L$, and $\forall t \in A : \sum_{a \in S_t} l_{at} < M$. Moreover, M is sufficiently smaller than $2^{\mathbb{k}}$, where $\mathbb{k}$ is the security parameter — the length in bits of the RSA modulus $\mathfrak{n}$ in the cryptographic keys of the agents (for example, $\mathbb{k} = 2048$, and $M = 2^{80}$).

Let $[0, X]$ be a large interval (for example, $[0, 2^{32} - 1]$).

To generate the zero-knowledge proof $setMembershipZKP(\beta_a, [(h_a \times M), (h_a \times M) + L])$ in step 10 of the event PREP, an agent $\boldsymbol{a}$ requires the randomization $\mathfrak{r}_{\beta_a}$ of the encryption β_a, which can be computed as follows: $\mathfrak{r}_{\beta_a} = \mathfrak{r}_{a,1} \times \ldots \times \mathfrak{r}_{a,k_a+1}$, where $\mathfrak{r}_{a,i}$ is the randomization used for the encryption of $E_a(x_{a,i})$.

To generate the zero-knowledge proof $plaintextEqualityZKP(\gamma_a, E_q(\sigma_a))$ in step 4 of the event VERIFIED_SHARES, an agent $\boldsymbol{a}$ requires the randomization $\mathfrak{r}_{\gamma_a}$ of the encryption γ_a, which can be computed as follows: $\mathfrak{r}_{\gamma_a} = (g^{-\sigma_a} \cdot \gamma_a)^{1/\mathfrak{n}_a \ mod \ (p-1)(q-1)} \ mod \ \mathfrak{n}_a$, where g and $\mathfrak{n}_a$ are in the public key of $\boldsymbol{a}$, and p and q are in the secret key, $\mathfrak{n}_a = pq$.

Table 5.1: Description of the functions used in Disruptive-Malicious-k-Shares.

Function	**Description**
timestamp()	Returns current time.
random(α,β)	Returns a random number uniformly distributed over the interval $[\alpha, \beta]$.
set_of_trustworthy($\boldsymbol{a}$, S)	Returns a set of agents $U_a = \{u_{a,1} \ldots u_{a,k_a}\}$, where $1 \leq k_a \leq k$, and $U_a \subseteq S$. The set U_a is selected such that: $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$ is low, with the minimum possible k_a.

Protocol: Disruptive-Malicious-k-Shares
Participants: Agents: q, t, and $S_t \equiv S_{t,\psi} = \{a_1 \dots a_n\}$. Agents q, t, and a subset of $S_{t,\psi}$ of size $m < n$ are considered to be dishonest, however, q wishes to learn the correct output (and therefore does not disrupt the protocol). $n \geq 3$.
Input: Each source agent $\boldsymbol{a}$ has a private input $l_{at} \equiv P(perform(a, t, \psi) = true)$.
Output: Agent q learns $r_{t,\psi}$, the reputation of agent t in the context ψ, or agent q learns the identity of the agents who disrupt the protocol.
Setup: Each agent $\boldsymbol{a}$ maintains $S_a \equiv S_{a,\psi}$, the set of its source agents in the context ψ. All communication takes place over authenticated point-to-point channels that are resistant to wire-tapping and tampering.
Events and Associated Actions (for an Agent $\boldsymbol{a}$):

need arises to determine r_t

▷ initiate query
1 send tuple (REQUEST_FOR_SOURCES, ψ) to t
2 receive tuple (SOURCES, ψ, S_t) from t
3 verify S_t from the source managers of t
4 retrieve public key PK_w of each agent $w \in S_t$ from a certificate authority
5 $S'_t \leftarrow S_t$ ▷ initialize the set of agents who are expected to send their shares
6 $\theta \leftarrow 0$ ▷ a cumulative sum for computing reputation
7 $V_w \leftarrow \phi$, for each agent $w \in S_t$ ▷ initialize the sets of encrypted shares
8 $s \leftarrow$ timestamp()
9 send tuple (PREP, q, t, s, S_t) to each agent $w \in S_t$

tuple (REQUEST_FOR_SOURCES, ψ) **received from agent** q

1 send tuple (SOURCES, ψ, S_a) to q

Figure 5.6: Protocol: Disruptive-Malicious-k-Shares

5.5.4 Security

Correctness

Theorem 9. *Let's assume that agent q is able to retrieve the correct S_t from the source managers of agent t. Then, in the protocol Disruptive-Malicious-k-Shares (Figure 5.6), agent q either learns the correct reputation of agent t in the context ψ, or it learns the identity of a malicious agent who has disrupted the protocol, under the disruptive malicious adversarial model.*

Proof. First, we consider the **semi-honest model**, which assumes that the protocol is followed according to the specification, and that the communication channels are not wire-tapped and not tampered with.

In the protocol Disruptive-Malicious-k-Shares (PREP: lines 3 – 5), each agent

Protocol: Disruptive-Malicious-k-Shares (contd.)

tuple $(\text{PREP}, q, t, s, S_t)$ **received from agent** q

▷ select trustworthy agents
1 $U_a \leftarrow$ set_of_trustworthy($\boldsymbol{a}$, $S_t - \{a\}$)
2 $k_a \leftarrow |U_a|$
▷ prepare shares
3 **for** $i \leftarrow 1$ **to** k_a
4 **do** $x_{a,i} \leftarrow \text{random}(0, X)$
5 $x_{a,k_a+1} \leftarrow (l_{at} - \sum_{i=1}^{k_a} x_{a,i})\ mod\ M$
6 $h_a \leftarrow (\sum_{i=1}^{k_a+1} x_{a,i})\ div\ M$
▷ retrieve public keys
7 retrieve the public key of each $u \in U_a$ and the public key of q from a certificate authority
▷ encrypt shares
8 encrypt $x_{a,1} \ldots x_{a,k_a+1}$ with the public key of $\boldsymbol{a}$ to obtain $E_a(x_{a,1}) \ldots E_a(x_{a,k_a+1})$ respectively
9 encrypt $x_{a,1} \ldots x_{a,k_a}$ with the public key of $u_{a,1} \ldots u_{a,k_a}$ to obtain $E_{u_{a,1}}(x_{a,1}) \ldots E_{u_{a,k_a}}(x_{a,k_a})$ respectively
▷ generate zero-knowledge proofs
10 $\beta_a \leftarrow (E_a(x_{a,1}) \times \ldots \times E_a(x_{a,k_a+1})) \bmod \mathfrak{n}_a^2$
11 generate setMembershipZKP($\beta_a, [(h_a \times M), (h_a \times M) + L]$)
12 **for** $i \leftarrow 1$ **to** k_a
13 **do** generate plaintextEqualityZKP($E_a(x_{a,i}), E_{u_{a,i}}(x_{a,i})$)
▷ send the encrypted shares and the proofs to q
14 $\overrightarrow{\mathcal{I}_a} \leftarrow \langle k_a, U_a, E_a(x_{a,1}), \ldots, E_a(x_{a,k_a+1}), E_{u_{a,1}}(x_{a,1}), \ldots, E_{u_{a,k_a}}(x_{a,k_a}),$
h_a, setMembershipZKP($\beta_a, [(h_a \times M), (h_a \times M) + L]$),
plaintextEqualityZKP($E_a(x_{a,1}), E_{u_{a,1}}(x_{a,1})$), $\ldots$,
plaintextEqualityZKP($E_a(x_{a,k_a}), E_{u_{a,k_a}}(x_{a,k_a})$)$\rangle$
15 send tuple $(\text{SHARES}, q, t, s, \overrightarrow{\mathcal{I}_a})$ to agent q

Figure 5.7: Protocol: Disruptive-Malicious-k-Shares (contd.)

Protocol: Disruptive-Malicious-k-Shares (contd.)

tuple (SHARES, $q, t, s, \overrightarrow{\mathcal{I}_v}$) **received from an agent** $v \in S_t$

▷ verify the set membership proof
1 $(\beta_v \leftarrow E_v(x_{v,1}) \times \ldots \times E_v(x_{v,k_v+1}))$ mod $\mathfrak{n}_v^2$
2 verify setMembershipZKP$(\beta_v, [(h_v \times M), (h_v \times M) + L])$
▷ verify the plaintext equality proofs
3 **for** $i \leftarrow 1$ **to** k_a
4 **do** verify plaintextEqualityZKP$(E_v(x_{v,i}), E_{u_{v,i}}(x_{v,i}))$
▷ manage the sets of encrypted shares to be relayed
5 **for** $i \leftarrow 1$ **to** k_a
6 **do** $V_{u_{v,i}} \leftarrow V_{u_{v,i}} \cup E_{u_{v,i}}(x_{v,i})$
▷ subtract v from the set of agents who are yet to send their shares
7 $S'_t \leftarrow S'_t - \{v\}$
▷ if shares have been received from all source agents then relay the shares
8 **if** $S'_t = \phi$
9 **then** $S'_t \leftarrow S_t$ ▷ initialize the set of agents who are yet to send their sum
10 send tuple (VERIFIED_SHARES, q, t, s, V_w) to each agent $w \in S_t$

tuple (VERIFIED_SHARES, q, t, s, V_a) **received from agent** q

▷ compute sum of the shares
1 $\gamma_a \leftarrow ((\prod_{c \in V_a} c) \times E_a(x_{a,k_a+1}))$ mod $\mathfrak{n}_a^2$
2 $\sigma_a \leftarrow D_a(\gamma_a)$
▷ encrypt the sum
3 encrypt σ_a with the public key of q to obtain $E_q(\sigma_a)$
▷ generate zero-knowledge proof
4 generate plaintextEqualityZKP$(\gamma_a, E_q(\sigma_a))$
▷ send the encrypted sum and the proof to agent q
5 send tuple (AGGREGATE, $q, t, s, E_q(\sigma_a), pezkp(\gamma_a, E_q(\sigma_a))$ to q

Figure 5.8: Protocol: Disruptive-Malicious-k-Shares (contd.)

Protocol: Disruptive-Malicious-k-Shares (contd.)

tuple (AGGREGATE, $q, t, s, E_q(\sigma_v), pezkp(\gamma_v, E_q(\sigma_v))$ **received from an agent** $v \in S_t$

▷ verify the proof
1 $\gamma_v \leftarrow ((\prod_{c \in V_v} c) \times E_v(x_{v,k_a+1}))$ mod $\mathfrak{n}_v^2$
2 verify plaintextEqualityZKP$(\gamma_v, E_q(\sigma_v))$
▷ decrypt the sum
3 $\sigma_v \leftarrow D_q(E_q(\sigma_v))$
▷ compute intermediate sum for reputation
4 $\theta \leftarrow \theta + \sigma_v$
▷ subtract v from the set of agents who are yet to send their sum
5 $S'_t \leftarrow S'_t - \{v\}$
▷ if sum has been received from all source agents, compute reputation
6 **if** $S'_t = \phi$
7 **then** $r_{t,\psi} \leftarrow (\theta \ mod \ M)/n$

Figure 5.9: Protocol: Disruptive-Malicious-k-Shares (contd.)

Table 5.2: Description of the variables used in Disruptive-Malicious-k-Shares.

Variable	Description
A	The set of all agents in the environment
a	A source agent. $a \in S_t$.
c	A ciphertext
h_a	The quotient when the sum of the shares of an agent $\boldsymbol{a}$ is divided by M. $h_a = (\sum_{i=1}^{k_a+1} x_{a,i})\ div\ M$.
$\overrightarrow{\mathcal{I}_a}$	A vector that contains the encrypted shares and the proofs sent by an agent $\boldsymbol{a}$ to the agent q
k_a	The cardinality of the set U_a. $k_a = \lvert U_a \rvert$.
$\mathbb{k}$	The security parameter. The length in bits of the RSA modulus $\mathbb{n}$ in the cryptographic keys of the agents. For example, $\mathbb{k} = 2048$.
L	A positive integer constant. $l_{at} \in [0, L]$. For example, $L = 10$.
l_{at}	The feedback of a source agent $\boldsymbol{a}$ about a target agent t
M	A publicly known modulus. $M > L$. $\forall t \in A : \sum_{a \in S_t} l_{at} < M$. $M \ll 2^{\mathbb{k}}$. For example, $\mathbb{k} = 2048$, $M = 2^{80}$.
m	The number of dishonest source agents in S_t. $m < n$.
n	The cardinality of the set S_t. $n = \lvert S_t \rvert$.
$\mathbb{n}$	The RSA modulus in the public key of an agent
$\mathbb{n}_a$	The RSA modulus in the public key of an agent $\boldsymbol{a}$
PK_a	The public key of an agent $\boldsymbol{a}$
q	The querying agent
$r_t \equiv r_{t,\psi}$	The reputation of an agent t in the context ψ
$S_t \equiv S_{t,\psi}$	The set of source agents of agent t in the context ψ
S'_t	An intermediate set that is initialized to S_t. The set of agents who are expected to send their shares and sums to agent q.
s	A timestamp
t	The target agent
U_a	The set of fellow source agents that an agent $\boldsymbol{a}$ selects as trustworthy
u	A source agent. $u \in S_t$.
V_w	The set of encrypted shares that agent q receives from other agents and then relays to agent w
v	A source agent. $v \in S_t$.
w	A source agent. $w \in S_t$.
X	A large positive integer constant. $x_{a,i} \in [0, X]$. For example, $X = 2^{32} - 1$.
$x_{a,i}$	The i^{th} share of an agent $\boldsymbol{a}$
β_a	The encrypted sum of an agent $\boldsymbol{a}$'s shares. $\beta_a = E_a(\sum_{i=1}^{k_a+1} x_{a,i})$.
γ_a	The encrypted sum of the shares received by an agent $\boldsymbol{a}$ and agent $\boldsymbol{a}$'s $k_a + 1$'th share x_{a,k_a+1}
θ	A cumulative sum for computing reputation
σ_a	The sum of the shares received by an agent $\boldsymbol{a}$ and agent $\boldsymbol{a}$'s $k_a + 1$'th share x_{a,k_a+1}
ψ	An action. The context for trust.

$a \in S_t$ prepares the shares $x_{a,1} \ldots x_{a,k_a+1}$, such that:

$$\sum_{i=1}^{k_a+1} x_{a,i} = (l_{at} + (h_a \times M)) \ mod \ M \tag{5.1}$$

The sum of all shares of all source agents may be given as:

$$\sum_{a \in S_t} \sum_{i=1}^{k_a+1} x_{a,i} = \sum_{a \in S_t} ((l_{at} + (h_a \times M)) \ mod \ M) \tag{5.2}$$

In the protocol (PREP: line 15, SHARES: line 10), each agent $a \in S_t$ sends its share $x_{a,i}$ to another agent in S_t through q, where $i \in \{1 \ldots k_a\}$. Each agent $a \in S_t$ computes σ_a, which is the sum of the received shares and its own k_a+1'th share (VERIFIED_SHARES: lines 1 – 2). We deduce that $\sum_{a \in S_t} \sigma_a$ is the sum of all shares received by all agents (that is, $\sum_{a \in S_t} \sum_{i=1}^{k_a} x_{a,i}$) and all $k_a + 1$'th shares (that is, $\sum_{a \in S_t} x_{a,k_a+1}$).

$$\sum_{a \in S_t} \sigma_a = \sum_{a \in S_t} \sum_{i=1}^{k_a} x_{a,i} + \sum_{a \in S_t} x_{a,k_a+1} \tag{5.3}$$

$$= \sum_{a \in S_t} \sum_{i=1}^{k_a+1} x_{a,i} \tag{5.4}$$

In the protocol, θ is computed as follows (*need arises to determine* r_t: line 6, AGGREGATE: line 4):

$$\theta = \sum_{a \in S_t} \sigma_a \tag{5.5}$$

From equations 5.2, 5.3, and 5.5:

$$\theta = \sum_{a \in S_t} ((l_{at} + (h_a \times M)) \ mod \ M) \tag{5.6}$$

$$(\theta \ mod \ M)/n = ((\sum_{a \in S_t} ((l_{at} + (h_a \times M)) \ mod \ M)) \ mod \ M)/n \tag{5.7}$$

Since $l_{at} \leq L < M$ for each $a \in S_t$, and $\sum_{a \in S_t} l_{at} < M$, we get:

$$(\theta \ mod \ M)/n = (\sum_{a \in S_t} l_{at})/n \tag{5.8}$$

In the protocol (AGGREGATE: line 7), agent q learns the reputation of agent t in the context ψ as:

$$r_{t,\psi} = (\theta \ mod \ M)/n \tag{5.9}$$

From equations 5.8, 5.9, and equation 3.2, we conclude that agent q learns the correct reputation of agent t in the context ψ under the semi-honest model.

Now, we consider the **disruptive malicious model**.

Malicious agents may 1) refuse to participate in the protocol, 2) prematurely abort the protocol, 3) selectively drop messages that they are supposed to send, 4) tamper with the communication channels, 5) wiretap the communication channels, and 6) provide incorrect information (for example, provide out of range values as their inputs, make incorrect computations).

Agent q***.***

Agent q wishes to learn the correct output therefore it would not take any of the actions 1 to 4, and 6. Wiretapping the communication channels has no effect on the correctness of the protocol.

Each Agent $a \in S_t$***.***

Each agent $a \in S_t$ communicates exclusively with agent q. If an agent $\boldsymbol{a}$ takes any of the actions 1 to 3, it would be exposed as malicious to agent q. *Note:* Agent q can then remove the malicious agent from the set of source agents and restart the protocol. Eventually, only those agents who do not take actions 1 to 3 will remain in the set of source agents.

An agent $a \in S_t$ is unable to tamper with the communication channels since we assume that all communication takes place over authenticated point-to-point channels that are resistant to tampering. Since each agent $a \in S_t$ communicates exclusively with agent q, it will be exposed as malicious if it does not conform to these requirements.

Wiretapping the communication channels has no effect on the correctness of the protocol.

The first tuple of information that an agent $a \in S_t$ provides agent q is: $(\text{SHARES}, q, t, s, \overrightarrow{\mathcal{I}_a})$, where $\overrightarrow{\mathcal{I}_a} = \langle\ U_a,\ E_a(x_{a,1}),\ \ldots,\ E_a(x_{a,k_a+1}),\ E_{u_{a,1}}(x_{a,1}),\ \ldots,\ E_{u_{a,k_a}}(x_{a,k_a}),\ \text{setMembershipZKP}(\beta_a, L),\ \text{plaintextEqualityZKP}(E_a(x_{a,1}), E_{u_{a,1}}(x_{a,1})),\ \ldots,\ \text{plaintextEqualityZKP}(E_a(x_{a,k_a}),\ E_{u_{a,k_a}}(x_{a,k_a}))\ \rangle$.

The correctness of the first four elements of the tuple and the set U_a can be trivially verified by agent q. The remaining information pertains to the shares prepared by agent $\boldsymbol{a}$. The shares have been prepared correctly if the following conditions hold true: 1) the shares add up to a value in $[(h \times M), (h \times M) + L]$; 2) $E_{u_{a,1}}(x_{a,1}), \ldots, E_{u_{a,k_a}}(x_{a,k_a})$ encrypt the same shares as $E_a(x_{a,1}), \ldots, E_a(x_{a,k_a})$ respectively; 3) $E_{u_{a,1}}(x_{a,1}), \ldots, E_{u_{a,k_a}}(x_{a,k_a})$ are encrypted with the public keys of agents $u_{a,1} \ldots u_{a,k_a}$ respectively.

The first condition holds true for an agent $\boldsymbol{a}$ if the verification of $\text{setMembershipZKP}(\beta_a, [(h_a \times M), (h_a \times M) + L])$ by agent q is successful. Agent q can verify the proof since it can independently compute β_a (due to the additive homomorphic property of the cryptosystem), L and M are publicly known, and h_a is provided by agent $\boldsymbol{a}$. An incorrect value of h_a will result in failure of the verification of the zero-knowledge proof. A zero-knowledge proof that shows membership in an interval with an incorrect h_a has no effect on the final output of the protocol since it is computed as $mod\ M$.

The second and third conditions hold true for an agent $\boldsymbol{a}$ if the verification of

each plaintextEqualityZKP($E_a(x_{a,i})$, $E_{u_{a,i}}(x_{a,i})$) by agent q is successful, where $i \in \{1 \dots k_a\}$. Agent q can verify these proofs since it can independently retrieve the public keys of agents $\boldsymbol{a}$ and $u_{a,1} \dots u_{a,k_a}$ from a certificate authority.

If the verification of the one set-membership zero-knowledge proof and the k_a plaintext-equality zero-knowledge proofs provided by an agent $\boldsymbol{a}$ succeeds, it implies that agent $\boldsymbol{a}$ has provided correct information pertaining to the shares that it prepared. Otherwise, agent $\boldsymbol{a}$ has provided incorrect information and it may therefore be considered as malicious.

The second tuple of information that an agent $a \in S_t$ provides agent q is: (AGGREGATE, q, t, s, $E_q(\sigma_a)$, $pezkp(\gamma_a, E_q(\sigma_a))$.

The correctness of the first four elements of the tuple can be trivially verified by agent q. The remaining information pertains to the sum σ_a. The sum has been computed correctly if the following condition holds true: γ_a and $E_q(\sigma_a)$ encrypt the same plaintext.

The condition holds true for an agent $\boldsymbol{a}$ if the verification of $pezkp(\gamma_a, E_q(\sigma_a))$ by agent q is successful. Agent q can verify the proof since it can independently compute γ_a (due to the additive homomorphic property of the cryptosystem) and it can independently retrieve the public key of agents $\boldsymbol{a}$ from a certificate authority.

If the verification of the plaintext-equality zero-knowledge proof provided by an agent $\boldsymbol{a}$ succeeds, it implies that agent $\boldsymbol{a}$ has provided correct information pertaining to the sum σ_a. Otherwise, agent $\boldsymbol{a}$ has provided incorrect information and it may therefore be considered as malicious.

Agent t.

We assume that agent q is able to retrieve the correct S_t from the source managers of agent t.

It follows that in the protocol Disruptive-Malicious-k-Shares (Figure 5.6), agent q either learns the correct reputation of agent t in the context ψ, or learns the identity of a malicious agent who has disrupted the protocol, under the disruptive malicious adversarial model. □

Privacy

Theorem 10. *Let's assume that for each agent $a \in S_t$ in the Disruptive-Malicious-k-Shares protocol (Figure 5.6), $P(perform(a, u_{a,1}, \rho) = false) \times \dots \times P(perform(a, u_{a,k_a}, \rho) = false)$ is low. Let's assume that agent q is able to retrieve the correct S_t from the source managers of agent t. Then Disruptive-Malicious-k-Shares is a real privacy preserving protocol under the disruptive malicious model.*

Proof. First, we consider the **semi-honest model**, which assumes that the protocol is followed according to the specification, and that the communication channels are not wire-tapped and not tampered with. The proof for this model is quite similar to the proof of privacy for the protocol semi-honest-k-shares.

Let's consider an agent $a \in S_t$. Agent $\boldsymbol{a}$ prepares the shares $x_{a,1} \dots x_{a,k_a+1}$ of its secret feedback value l_{at}. The first k_a shares $x_{a,1} \dots x_{a,k_a}$ are random

numbers uniformly distributed over a large interval. The final share, $x_{a,k_a+1} = (l_{at} - \sum_{i=1}^{k_a} x_{a,i})\ mod\ M$, is also a number uniformly distributed over a large interval since it is a function of the first k_a shares which are random numbers. Thus, individually each of the shares does not reveal any information about the secret feedback value l_{at}. Moreover, no information is learnt about l_{at} even if up to k_a shares are known, since their sum would be some random number uniformly distributed over a large interval. The only case in which information can be gained about l_{at} is if all $k_a + 1$ shares are known. Then, $l_{at} = (\sum_{i=1}^{k_a+1} x_{a,i})\ mod\ M$.

We now analyze if the $k_a + 1$ shares of an agent $\boldsymbol{a}$ can be learnt by the adversary from the protocol.

Agent $\boldsymbol{a}$ sends each share $x_{a,i}$ only to agent $u_{a,i}$, where $i \in \{1 \ldots k_a\}$. Although, agent q relays each share $x_{a,i}$ from agent $\boldsymbol{a}$ to agent $u_{a,i}$, agent q or any third agent is unable to learn the share $x_{a,i}$ since it is sent encrypted with agent $u_{a,i}$'s public key as $E_{u_{a,i}}(x_{a,i})$. Only agent $u_{a,i}$ is able to decrypt $E_{u_{a,i}}(x_{a,i})$ and obtain $x_{a,i}$.

Each agent $u_{a,i}$ computes $\sigma_{u_{a,i}}$, which is the sum of all shares that it receives and its own final share $x_{u_{a,i},k_{u_{a,i}}+1}$. Even if agent $\boldsymbol{a}$ is the only agent to send agent $u_{a,i}$ a share, $\sigma_{u_{a,i}} = x_{a,i} + x_{u_{a,i},k_{u_{a,i}}+1}$. That is, the sum of agent $\boldsymbol{a}$'s share and agent $u_{a,i}$'s final share. Consequently, $\sigma_{u_{a,i}}$ is a number uniformly distributed over a large interval. Thus, when agent $u_{a,i}$ sends this number to agent q, it is impossible for q to distinguish the individual shares from the number. Therefore, each share $x_{a,i}$ that agent $\boldsymbol{a}$ sends to agent $u_{a,i}$ will only be known to agent $u_{a,i}$. Unless, agent $u_{a,i}$ is dishonest. The probability that agent $u_{a,i}$ is dishonest, that is, it will attempt to breach agent $\boldsymbol{a}$'s privacy is given as: $P(perform(a, u_{a,i}, \rho) = false)$.

To learn the first k_a shares of agent $\boldsymbol{a}$, all agents $u_{a,1} \ldots u_{a,k_a}$ would have to be dishonest. The probability of this scenario is given as: $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$.

Even in the above scenario, the adversary does not gain information about l_{at}, without the knowledge of agent $\boldsymbol{a}$'s final share x_{a,k_a+1}. However, agent $\boldsymbol{a}$ has to send $\sigma_a = x_{a,k_a+1} + \sum_{v \in J_a} x_v$, and agent $\boldsymbol{a}$ has no control over the $\sum_{v \in J_a} x_v$ portion of the equation. Therefore, we assume that agent q learns the final share of agent $\boldsymbol{a}$.

Thus the probability that the protocol will not preserve agent $\boldsymbol{a}$'s privacy can be stated as: $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$. We assume that the agents $u_{a,1} \ldots u_{a,k_a}$ are selected such that this probability is low. Therefore, with high probability, the adversary learns no more information about l_{at} than it can learn in the ideal protocol with what it knows before the execution of the protocol and the outcome.

Now, we consider the **disruptive malicious model**. Malicious agents may 1) refuse to participate in the protocol, 2) prematurely abort the protocol, 3) selectively drop messages that they are supposed to send, 4) tamper with the communication channels, 5) wiretap the communication channels, and 6) provide incorrect information (for example, provide out of range values as their

inputs).

Privacy of Agent q ***and Agent*** t***.***

Agent q and agent t are not required to contribute any private information during the protocol.

Privacy of Each Agent $a \in S_t$***.***

Attack 1. Refuse to Participate in the Protocol.

If a source agent v refuses to participate in the protocol, it has no effect on the privacy of any agent $\boldsymbol{a}$ since agent v must receive a share of agent $\boldsymbol{a}$'s private information before it can attack its privacy.

Agent t's refusal to participate also has no effect on the protocol. Agent q may retrieve S_t directly from agent t's source managers.

Agent q does not refuse to participate in the protocol since it wishes to learn the correct output of the protocol.

Attack 2. Prematurely Abort the Protocol.

If a source agent v prematurely aborts the protocol before receiving the shares, it has no effect on the privacy of any agent $\boldsymbol{a}$ since agent v must receive a share of agent $\boldsymbol{a}$'s private information before it can attack its privacy. The other scenario is that the source agent v prematurely aborts the protocol after receiving a share of agent $\boldsymbol{a}$'s private information. In that case, all first k_a shares of agent $\boldsymbol{a}$ must still be known to breach $\boldsymbol{a}$'s privacy. Thus prematurely aborting the protocol does not give an agent $v \in S_t$ any advantage in learning agent $\boldsymbol{a}$'s private information.

If agent t aborts the protocol before providing S_t, agent q may retrieve S_t directly from agent t's source managers. Therefore agent t's disruption has no effect on the protocol.

Agent q has no incentive to prematurely abort the protocol since it wishes to learn the correct output of the protocol, which is not learnt until after the last step.

Attack 3. Selectively Drop Messages.

If a source agent v selectively drops messages or parts of messages, it has no effect on the condition that all first k_a shares of agent $\boldsymbol{a}$ must be known to breach an agent $\boldsymbol{a}$'s privacy. Thus this is another action that does not give an agent $v \in S_t$ any advantage in learning agent $\boldsymbol{a}$'s private information.

Agent t may not provide S_t or may provide only a subset, however, that has no effect on the protocol since q also retrieves and verifies S_t from agent t's source managers.

Agent q does not selectively drop messages since it wishes to learn the correct output of the protocol. *Note:* Please see the discussion on Attack 6 for the case where agent q may relay incorrect shares or may not relay them at all.

Attack 4. Tamper with the Communication Channels.

Any agent is unable to tamper with the communication channels since we assume that all communication takes place over authenticated point-to-point channels that are resistant to tampering.

Attack 5. Wiretap the Communication Channels.

Any agent is unable to wiretap the communication channels since we assume that all communication takes place over authenticated point-to-point channels

that are resistant to wiretapping.

Attack 6. Provide Incorrect Information.

If a source agent v provides incorrect information, that has no effect on the condition that all first k_a shares of agent $\boldsymbol{a}$ must be known to breach an agent $\boldsymbol{a}$'s privacy. Agent v provides no information to agent $\boldsymbol{a}$ or agent q that would result in agent $\boldsymbol{a}$ divulging any extra information.

Agent t may provide an incorrect S_t, however, that has no effect on the protocol since q also retrieves and verifies S_t from agent t's source managers.

Agent q sends two types of messages to source agents: $PREP$, and $VERIFIED\_SHARES$.

$PREP$: Agent q may create S_t itself in order to attack an agent $a \in S_t$. The set may be created such that it contains all dishonest agents except agent $\boldsymbol{a}$ who is under attack. However, we assume that $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$ is low. That is, there exist trustworthy agents in the protocol such that agent $\boldsymbol{a}$ receives a high enough privacy guarantee. *Note:* the protocol may be extended such that an agent $\boldsymbol{a}$ is allowed to abstain if the privacy guarantee is not sufficient. In that case, modifying S_t would not help agent q to breach agent $\boldsymbol{a}$'s privacy. The extension would be as follows: The agent who wishes to abstain would generate the shares such that their sum equals zero. The abstaining agent would inform the querying agent that it has abstained, and would prove that the sum of the shares equals zero. The effect of this extension on the correctness of the protocol is discussed in sections 5.5.6 and 4.5.1.

$VERIFIED\_SHARES$: Agent q may substitute the shares sent by other agents to an agent $\boldsymbol{a}$ with shares that it has created itself. Agent q may also not relay a share at all. In both these cases, the best outcome for q would be to learn agent $\boldsymbol{a}$'s $k_a + 1$'th share. This has no effect on the privacy of agent $\boldsymbol{a}$ since agent q is still unable to learn its first k_a shares. Each of those shares is encrypted and may only be decrypted by its destination agent.

The protocol Disruptive-Malicious-k-Shares is a real privacy preserving reputation protocol (Definition 20) under the disruptive malicious model, since: 1) Disruptive-Malicious-k-Shares has the same parameters as Disruptive-Malicious-Ideal (except the TTP), and 2) the adversary does not learn any more information about the private input of any agent $\boldsymbol{a}$ in Disruptive-Malicious-k-Shares than it can learn in Disruptive-Malicious-Ideal, with high probability $(1 - P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false))$, under the disruptive malicious adversarial model. □

5.5.5 Complexity

The protocol requires $O(n)$ messages to be exchanged. The protocol also performs a DHT lookup in the initiation phase, which requires an additional $O(log\ N)$ messages (assuming Chord). Thus the total number of messages exchanged is $O(n) + O(log\ N)$, where n is the number of source agents in the protocol and N is the total number of agents in the system respectively.

Table 5.3: Protocol Disruptive-Malicious-k-Shares – Complexity.

Tuple	**Occurrences**	**IDs**	**Numbers**
REQUEST_FOR_SOURCES	1		
SOURCES	1	n	
PREP	n	$n \times n = n^2$	
SHARES	n	kn	
VERIFIED_SHARES	n		
AGGREGATE	n		
Total	$4n + 2$	$n + n^2 + kn$	
Complexity	$O(n)$	$O(n^2)$, for $k \ll n$	$O(1)$

Table 5.4: Protocol Disruptive-Malicious-k-Shares – Complexity (contd.).

Tuple	**Ciphertexts**	**SMZKPs**	**PEZKPs**
REQUEST_FOR_SOURCES			
SOURCES			
PREP			
SHARES	kn	n	kn
VERIFIED_SHARES	kn		
AGGREGATE	n		n
Total	$2kn + n$	n	$kn + n$
Complexity	$O(n)$, for $k \ll n$	$O(n)$	$O(n)$, for $k \ll n$

In terms of bandwidth used, the protocol requires transmission of the following amount of information: $O(n^2)$ agent IDs, $O(n)$ ciphertexts, $O(n)$ non-interactive zero-knowledge proofs of set membership, and $O(n)$ non-interactive zero-knowledge proofs of plaintext equality.

This complexity analysis assumes that $k \ll n$. In Experiments 3 and 4 (sections 4.8.4 and 4.8.5) on the Advogato.org web of trust, we observed that $k \ll n$ is a reasonable assumption. The experiments show that the privacy of 81.7% of instances of source agents is preserved with $k = 2$ and $n \geq 25$. Similarly, the privacy of 85.8% of instances of source agents is preserved with $k = 2$ and $n \geq 50$. The maximum percentage of instances of source agents whose privacy can be preserved (even with $k = n$) is around 87%.

5.5.6 Discussion

Table 5.5 provides a comparison of our reputation protocol for the disruptive malicious model with the other systems in the literature.

Our protocol does not require centralized constructs or specialized hardware. The communication complexity of the protocol is $O(n) + O(log\ N)$, where n is the number of feedback providers and N is the number of all entities in the system.

In contrast, the privacy preserving reputation systems discussed in the lit-

erature for the disruptive malicious model, either rely on centralized constructs or specialized hardware, or require very high communication complexity.

The reputation systems by Androulaki et al. and Steinbrecher are very efficient. However, these systems rely on centralized constructs, which makes them unsuitable for decentralized networks. Kinateder et al.'s reputation system provides anonymity in peer-to-peer systems under the disruptive malicious model. However, the system functions only for specialized networks. The protocol by Pavlov et al. (based on their second witness selection scheme) is the only other reputation protocol that we have found that does not require centralized constructs or specialized networks. However, this protocol needs $O(n^3)$ messages to be exchanged, which makes it prohibitively expensive.

Table 5.5: Protocol Disruptive-Malicious-k-Shares – Comparison.

System / Protocol	**Architecture**	**Target Environment**	**Key Security Mechanisms**	**Privacy Guarantee**	**Complexity (Messages)**
Disruptive-Malicious-k-Shares	D	Distributed environments	Additive homomorphic cryptosystems, Zero-knowledge proofs	If $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$ is low for each $a \in S_t$	$O(n) + O(log\ N)$
Pavlov et al. [105] – WSS-2	D	Distributed environments	Verifiable secret sharing, discrete log commitment	If $b < \frac{N}{2} - n$	$O(n^3)$, where n = number of witnesses
Androulaki et al. [9]	D	Peer-to-peer systems	Anonymous credential systems, E-cash (bank), Blind signatures, Mixnets / Onion Routing	If the underlying primitives (anonymous credential system, e-cash system, and blind signatures) are secure	$O(1)$
Kinateder and Pearson [81]	D	Peer-to-peer systems	Trusted platform, MIX cascades, Digital signatures	If the underlying primitives (trusted platform, MIX cascades, digital signatures) are secure	Not Provided
Steinbrecher [126]	C	E-commerce, Self-help forums, etc.	Pseudonym / Identity management	If the provider (central server) is honest and secure	$O(1)$

Chapter 6

Trust Recommendation and Propagation

In this chapter, we focus on some issues related to trust recommendation and propagation. The concepts of trust recommendation and propagation (introduced in Section 2.1.3) are closely related to reputation as they are alternative methods for establishing trust.

Establishing trust in an unknown entity through trust recommendation and propagation takes advantage of the possible transitivity of trust. Let's say that Alice wishes to establish trust in an unknown individual Carol. If another individual Bob trusts Carol, then he could give a recommendation to Alice about Carol's trustworthiness. Taking Bob's trust recommendation and her own trust in Bob into account, Alice may establish a trust relationship with Carol. Thus a transitive path of trust that leads from Alice to Bob to Carol, enables Alice to develop trust in Carol. If Alice wishes to establish trust in Carol through Bob's recommendation, we say that Bob's trust in Carol has propagated to Alice.

The first issue that we address in this chapter is the effect of subjectivity on trust recommendation. We highlight the problem that when a trust value is recommended by one user to another, it may lose its real meaning due to subjectivity. As an example, consider that an agent Alice regards a trust value such as 0.8 as a very high value of trust. It is possible that another agent Bob perceives this same trust value as only average. If Bob conveys to Alice that his trust in an agent Carol is 0.8, Alice may misinterpret Carol as highly trustworthy, whereas according to Bob's belief Carol has only average trustworthiness. We present a solution based on the notion of percentiles for the elimination of subjectivity from trust recommendation. We run experiments to compare our subjectivity-eliminated trust recommendation method with the unmodified method that does not take subjectivity into account. It is observed that our method can give better results over 90% of the time.

The second issue that we explore in this chapter is related to one of the important applications of trust propagation, namely access control in multi-domain

environments. There are a number of models for access control in multi-domain environments that are based on trust propagation. Iterative multiplication of the trust values on a path from a querying entity to a target entity is one of the common strategies for trust propagation. We evaluate the effectiveness of this strategy. The data set used for this evaluation is the real web of trust of Advogato.org. We find that a substantial positive linear correlation exists between trust values based on direct experience and the corresponding propagated trust values derived through the iterative multiplication approach.

We also shed light on the problem of privacy as it relates to each of the two issues explored in this chapter.

6.1 Extensions to the General Framework

In this section, we extend the general framework, presented in Chapter 3, to accommodate trust recommendation and propagation. We differentiate between two types of trust propagation: simple (atomic) and compound (iterative) trust propagation.

For readability we include the definition of source agent from Chapter 3.

Definition. ***Source Agent.*** *An agent* $\boldsymbol{a}$ *is said to be a source agent of an agent* $\boldsymbol{b}$ *in the context of an action* ψ *if* $\boldsymbol{a}$ *has trust in* $\boldsymbol{b}$ *in the context* ψ*. In other words, agent* $\boldsymbol{a}$ *is a source agent of agent* $\boldsymbol{b}$ *in context* ψ *if* $\langle aTb, \psi, P(perform(a, b, \psi) = true)\rangle \in \mathbb{U}$.

Definition 27. ***Trust Recommendation.*** *Let an agent* $\boldsymbol{a}$ *be a source agent for an agent* t *in context* ψ*. Let an agent* q *request the source agent* $\boldsymbol{a}$ *for* $l_{at} \equiv P(perform(a, t, \psi) = true)$*. If agent* $\boldsymbol{a}$ *reports* l_{at} *to agent* q*, then the act of reporting the trust value is said to be a trust recommendation from* $\boldsymbol{a}$ *to* q *about* t*. Additionally, the reported trust value* l_{at} *is said to be a recommended trust value from* $\boldsymbol{a}$ *to* q *about* t.

Definition 28. ***Querying Agent (In the context of trust recommendation).*** *When an agent* q *requests a source agent* $\boldsymbol{a}$ *to report* $l_{at} \equiv P(perform(a, t, \psi) = true)$ *as a recommended trust value, then we refer to agent* q *as the querying agent.*

Definition 29. ***Target Agent (In the context of trust recommendation).*** *When an agent* q *requests a source agent* $\boldsymbol{a}$ *to report* $l_{at} \equiv P(perform(a, t, \psi) = true)$ *as a recommended trust value, then we refer to agent* t *as the target agent.*

Definition 30. ***Simple (Atomic) Trust Propagation.*** *Let an agent* q *be a querying agent who receives recommended trust* l_{at} *from a source agent* $\boldsymbol{a}$ *about a target agent* t*. Let's say that as a consequence of the trust recommendation, agent* q *establishes* $P(perform(q, t, \psi) = true) = l_{at}$*. Then this act of* q *establishing trust in* t *as a consequence of a trust recommendation from* $\boldsymbol{a}$ *is said to be a simple trust propagation from* $\boldsymbol{a}$ *to* q *about* t*. Additionally, the newly established trust value* $P(perform(q, t, \psi) = true)$ *is said to be simple propagated trust from* $\boldsymbol{a}$ *to* q *about* t.

Definition 31. ***Trust Path.*** *We define a trust path $p_\psi(a_1, a_n)$ from an agent a_1 to an agent a_n as a sequence of agents $\langle a_1 \dots a_n \rangle$ such that trust $\langle a_{i-1}\ T\ a_i, \psi, P(perform(a_{i-1}, a_i, \psi) = true) \rangle \in \mathbb{U}$ for $i = 2 \dots n$. The length of a trust path $p_\psi(a_1, a_n)$ is the number of agents in the sequence $\langle a_1 \dots a_n \rangle$, which is n in this case.*

In other words, a trust path is a sequence of agents such that each agent has trust in its successor agent in a context ψ.

Definition 32. ***Function ptrust.*** *Let $p_\psi(a_1, a_n)$ be a trust path from an agent a_1 to an agent a_n. Then, $ptrust(p_\psi(a_1, a_n))$ is defined as a function, such that: $ptrust$: trust path $\rightarrow [0, 1]$.*

The function *ptrust* (abbreviation of "propagated trust"), given a path $\langle a_1, \dots, a_n \rangle$, computes a weight for the edge (a_1, a_n).

Definition 33. ***Compound Trust Propagation.*** *Let an agent a_1 be a querying agent who receives recommended trust $l_{a_2 a_3} \dots l_{a_{n-1} a_n}$ in context ψ from source agents $a_2 \dots a_{n-1}$ respectively about target agents $a_3 \dots a_n$ respectively. Let's say that as a consequence of the trust recommendations, agent a_1 establishes $P(perform(a_1, a_n, \psi) = true) = ptrust(p_\psi(a_1, a_n))$, where a trust path $p_\psi(a_1, a_n)$ exists from agent a_1 to agent a_n. Then this act of a_1 establishing trust in a_n as a consequence of the trust recommendations by agents $a_2 \dots a_{n-1}$ is said to be compound trust propagation. Additionally, the newly established trust value $P(perform(a_1, a_n, \psi) = true)$ is said to be compound propagated trust.*

6.2 Subjectivity in Trust Recommendation

The quantification of trust by agents is subjective, therefore two agents may assign different trust values to a third party even if it exhibits similar behavior towards both. We argue that this subjectivity bears an undesirable effect on trust recommendation. A querying agent and a source agent may associate very different meanings with a particular trust value. Thus the true meaning of a trust value may not be conveyed in a trust recommendation due to subjectivity.

Let's consider the following scenario: Alice and Bob are two online shoppers and Carol is an online vendor. The context for Carol's trustworthiness is the action "deliver product on time". Alice and Bob both place an order with Carol who delivers the product 2 days late to each of them with an assurance that the delay will not occur again. In this scenario, the behavior of Carol is the same towards both Alice and Bob, however, each of them will quantify their trust subjectively according to their own perception. Bob who is trusting might believe Carol's assurance and quantify his trust as let's say 70% (subjective probability). Whereas, Alice who is less trusting might be inclined towards a lower value such as 40%. Imagine if Alice had not interacted with Carol and received a recommendation of 70% about her from Bob. She would clearly misinterpret Carol as being someone with higher trustworthiness. In the following

section we observe that this issue is not limited to the quantification of trust as subjective probability.

Several works [26], [60], [98] propose trust models that aim to capture the subjectivity aspect of human trust. However, the focus is on enabling agents to form trust opinions that are uniquely their own in contrast to delegating trust formation to some external authority. None of the cited works address subjectivity as it affects trust recommendation. In this thesis, we focus specifically on the problem of subjectivity in trust recommendation. We are interested in developing a method for trust recommendation in which subjectivity causes lesser distortion in the meaning of the recommended trust value.

6.2.1 Trust Representation and Subjectivity

How does one represent the amount of trust that one agent associates with another? A common approach is to represent the spectrum of trust quantitatively as a numerical range. Marsh's formalism [90] represents trust as a continuous variable over an interval of $[-1,1]$. Golbeck's FilmTrust [54] defines an integer range of 1 to 10. Gambetta [52], Griffiths [60], and Toivonen [131] utilize an interval of [0,1] for the purpose.

An alternate approach is to divide the span of trust into strata and assign them qualitative labels. The stratification used by Abdul-Rahman and Hailes [1] is given as the set {Very Trustworthy, Trustworthy, Untrustworthy, Very Untrustworthy}. Jonker and Treur [75] use a similar stratification defined as the ordering: Unconditional Distrust $<$ Conditional Distrust $<$ Conditional Trust $<$ Unconditional Trust. Levien's Advogato [86] allows users to rate each other as Observer (minimum trust), Apprentice, Journeyer, or as Master (maximum trust).

Consider a scenario where Bob assigns a trust value of 0.8 to Carol on an interval of [0,1] with 1 representing maximum trust. Let's assume that 0.8 is an average trust value if it is viewed in the context of trust values that Bob has assigned to other entities. Thus Bob perceives Carol as someone being moderately trustworthy. With whatever skew Bob assigns trust values to other entities, it presents no problem inside his local environment since all those values occur in the same context.

The problem of subjectivity arises when Bob conveys to Alice that his trust in Carol is represented by the value 0.8. It is likely that a value of 0.8 signifies something very different to Alice. Is 0.8 an average value of trust for Alice as was the case for Bob? Or is 0.8 a very high value of trust for Alice? Given the context of Alice's history of trust value assignments, we may discover that Alice rarely assigns a value of 0.8 to any entity and thus associates very high trust with such a value. In Bob's position Alice might have assigned a value such as 0.6 to Carol. Alice may make a misjudgment of Carol's trustworthiness if she bases her decision on her own perception of the trust value conveyed to her by Bob. We observe that due to subjectivity, the meaning of a trust value is distorted when it is transferred between agents. Subjectivity occurs due to differences in dispositions to trust. Disposition to trust is defined and discussed

in the next section.

We subscribe to the definition of subjectivity given by the Merriam-Webster dictionary (merriam-webster.com) as a judgment that is "modified or affected by personal views, experience, or background" and is "peculiar to a particular individual".

The use of strata with qualitative labels may initially be considered as a solution to the problem of subjectivity. We may argue that a stratified trust representation model, such as the four distinct strata defined by Abdul-Rahman and Hailes [1], provides clear semantics and avoids the ambiguity associated with numerical values. The reasoning being that a qualitative label such as "trustworthy" should hold the same meaning for one entity as it does for another.

However, we concur with Griffiths [60] and Marsh [90] that the stratification approach also suffers from the problem of subjectivity. Different entities may associate the same experiences with different strata. For example, based on their own perception of trust, what is viewed by Bob as "very trustworthy" may be judged as merely "trustworthy" by Alice.

We further observe that subjectivity is an issue when the resolution of the scale for quantifying trust is high. Let's consider the boolean scale $\{0, 1\}$, where 0 represents "not trustworthy" and 1 represents "trustworthy". Such a scale allows little room for misinterpretation as compared to a scale with higher resolution such as $\{1 \ldots 10\}$ or $[0, 1]$.

6.2.2 Disposition to Trust

Disposition to trust is the inherent propensity of an individual to trust or distrust others. An individual's disposition to trust does not vary for specific entities but is a stable characteristic of their personality that governs how they view the trustworthiness of every other entity that they encounter.

McKnight et al [95] define disposition to trust as the "extent to which a person displays a tendency to be willing to depend on others across a broad spectrum of situations and persons".

Rotter [116], [117] notes that an individual's "generalized attitude" towards trust is a product of life experiences, such as interactions with parents, peers, and authorities. Boone and Holmes [19] suggest that good experiences lead to a greater disposition to trust and vice versa.

A study in the context of ecommerce by McCord and Ratnasingam [93] has demonstrated that there is a strong relationship between an individual's disposition to trust and the trust related decisions that they make.

A thorough treatment of the literature on disposition to trust is provided by Kaluscha [79].

We now revisit Alice, Bob, and Carol from our previous example. Alice and Bob are two individuals with different dispositions to trust. Bob has a high disposition to trust and thus assigns a high trust value of 0.8 to Carol. In contrast, Alice who has a lower disposition to trust, rates Carol's trustworthiness as only 0.6. This subjectivity occurs despite the fact that Carol exhibits the same behavior in her interactions with both Alice and Bob.

Quantitative Representation of an Agent's Disposition to Trust

The method that we present in the next section requires quantitative representation of the disposition to trust of agents. We discuss three possible alternatives for this purpose.

Manually specified by the agent. The agent may be presented with a scale, for example, 1 to 10 or [0,1] and asked to rate their disposition to trust manually. The approach is simple and straightforward. However, the disadvantage of this approach is that the agent has to be explicitly engaged by the process. Moreover, it is debatable if an agent himself is a true judge of his own disposition to trust.

Assessed through a trust scale. A number of researchers have developed trust scales that help assess the disposition to trust of a person. The subject is required to respond to a series of questions with weighted multiple choice answers. The cumulative score of the subject indicates their disposition to trust.

Rotter's Interpersonal Trust Scale [117] and Christie and Geis's Machiavellianism Scale [30] are examples of this approach. A sample question from Rotter's Interpersonal Trust Scale is as follows:

> "In dealing with strangers one is better off to be cautious until they have provided evidence that they are trustworthy."
> *Answer choices:* strongly agree (weight: 1), mildly agree (2), agree and disagree equally (3), mildly disagree (4), strongly disagree (weight: 5).

Rotter's and the Machiavellianism trust scales are likely to assess the disposition to trust of an individual accurately. However, the requirement that each agent make themselves available for a series of questions discounts their practicality.

Inferred from an agent's history of trust value assignments. Several examples from computer science literature may be cited where historical patterns are used to predict future behavior with considerable success. Instances include Self-Customizing Software [68] or Adaptive User Interfaces [85], and Branch Predictors in Microprocessors [50].

We propose an approach based on similar lines for determining the disposition to trust of an agent. The trust values that an agent has assigned in the past may be considered as an indication of their disposition to trust. For example, given an agent who has a pattern of assigning high values of trust, we may infer that the agent has a high disposition to trust, and vice versa. We thus propose to represent an agent's disposition to trust by the collection of their previous trust value assignments in a system.

A close approximation of an agent's disposition to trust is possible only if they have made a significant number of trust value assignments in the

past. The question is what number can be considered as significant. We attempt to address this question in the experiments.

A key reason why we choose this approach for the representation of disposition to trust is that it does not require additional input from an agent. This also implies that given a web of trust, we can test our method without requiring each agent to explicitly establish their disposition to trust.

Definition 34. ***Disposition to Trust.*** *Let a set $O_{u,\psi} = \{v \mid$ trust of u in v in the context ψ exists in $\mathbb{U}\}$. In other words, $O_{u,\psi}$ is the set of all agents for whom agent u is a source agent in the context ψ. Let a vector $o_{u,\psi} = \langle l_{uv_1} \ldots l_{uv_{n_{u,\psi}}} \rangle$, where $v_i \in O_{u,\psi}$, $n_{u,\psi} = |O_{u,\psi}|$, and $l_{uv_i} \equiv P(perform(u, v_i, \psi) = true)$. The vector $o_{u,\psi}$ comprises of all trust values that agent u has assigned to other agents in the context ψ. The disposition to trust of agent u in the context ψ is given as the vector $d_{u,\psi} = sort(o_{u,\psi})$, where function sort returns a vector which comprises of the elements of $o_{u,\psi}$ sorted in ascending order. The elements of $d_{u,\psi}$ are given as $d_{u,\psi,j}$, where the index $j \in \{1 \ldots n_{u,\psi}\}$.*

Definition 35. ***Function first.*** *Let $d_{u,\psi}$ be the disposition to trust of an agent u in the context ψ. Let l_{uv} be an element in $d_{u,\psi}$, such that $v \in O_{u,\psi}$. Then, $first(l_{uv}, d_{u,\psi})$ is defined as a function that outputs the lowest index (the index of the first occurrence) of the element l_{uv} in $d_{u,\psi}$.*

6.2.3 A Method for Elimination of Subjectivity from Trust Recommendation

In this section we introduce our method for the elimination of subjectivity from trust recommendation.

As we have discussed earlier, the trust values assigned by an agent are subjective to its disposition to trust. When a source agent recommends a target agent, the meaning of the associated trust value is distorted due to the different disposition to trust of the querying agent.

The solution we propose is to report trust not as an absolute score but a value that is relative to the disposition to trust of the source agent. In other words, we report the relative standing of the source agent's trust in the target agent in terms of the entire trust value assignments that the source agent has made.

Two options for implementing this idea are reporting trust as either a standard score (z-score), or as a percentile. If we report as z-score, then it is required that the trust values assigned by agents be normally distributed. On the other hand if we report as percentile then there is no such requirement. We therefore opt for a solution based on percentiles.

A percentile value indicates the source agent's perception of the target agent in relation to the others that the source agent has rated.

Going back to the example (discussed in the opening of the chapter), if Bob conveys to Alice an absolute value such as 0.8, Alice does not know if according

to Bob, the value 0.8 is an average value or a very high value of trust. However, if the trust is reported as a percentile value, Alice does have this information. For example, if the percentile value is in the vicinity of 50%, Alice would know that according to Bob, Carol has an average trustworthiness. If the percentile value is around 80% or 90%, it is clear that Bob regards Carol as highly trustworthy. The absolute value that Bob locally assigned to Carol becomes irrelevant.

To convert the percentile to a local absolute score, the querying agent reads the value that is at the given percentile in the collection of trust values that he himself has assigned to other agents. This absolute score holds perfect meaning for the querying agent since it is in the context of his own disposition to trust.

Thus we attempt to eliminate the subjectivity and misinterpretation associated with an absolute value of trust by going through an intermediate relative value.

We note that this method does not require agents to make any modifications to the way they evaluate other agents. Locally, each agent establishes their trust beliefs as usual, in terms of their own disposition to trust. Another key aspect of this solution is that it does not require the involvement of any third parties or centralized entities and is therefore suitable for decentralized environments.

6.2.4 Formal Description of the Method

The implementation of the functions *percentile* and $fbvalue$ is based on the method for the estimation of percentiles given by NIST [100].

Definition 36. ***Function percentile.*** *$percentile(l_{uv}, d_{u,\psi})$ is defined as a function that outputs c_{uv}, the percentile of a feedback value l_{uv} in $d_{u,\psi}$, where $l_{uv} \equiv P(perform(u, v, \psi) = true)$ and $v \in O_{u,\psi}$.*

$$\begin{aligned} c_{uv} &= percentile(l_{uv}, d_{u,\psi}) \\ &= \frac{100 \cdot first(l_{uv}, d_{u,\psi})}{n_{u,\psi} + 1} \end{aligned}$$

where $n_{u,\psi} = |O_{u,\psi}|$.

Definition 37. ***Function fbvalue.*** *$fbvalue(c_{uv}, d_{w,\psi})$ is defined as a function that outputs $l_{uv:w}$, the feedback value at the c_{uv} percentile in $d_{w,\psi}$, where $c_{uv} = percentile(l_{uv}, d_{u,\psi})$.*

$$
\begin{aligned}
l_{uv:w} &= fbvalue(c_{uv}, d_{w,\psi}) \\
&= \begin{cases} d_{w,\psi,i} + f \cdot (d_{w,\psi,i+1} - d_{w,\psi,i}) & \text{if } 0 < i < n_{w,\psi} \\ d_{w,\psi,1} & \text{if } i = 0 \\ d_{w,\psi,n_{w,\psi}} & \text{if } i = n_{w,\psi} \end{cases}
\end{aligned}
$$

where,

$$i = \left\lfloor \frac{c_{uv} \cdot (n_{w,\psi} + 1)}{100} \right\rfloor$$

and,

$$f = \frac{c_{uv} \cdot (n_{w,\psi} + 1)}{100} - i$$

i is an integer and $0 \leq f < 1$.

Definition 38. ***Subjectivity Eliminated Trust Recommendation.*** *Let an agent $\boldsymbol{a}$ be a source agent for an agent t in the context ψ. Then an agent q may request the source agent $\boldsymbol{a}$ for $c_{at} = percentile(l_{at}, d_{a,\psi})$. If $\boldsymbol{a}$ reports c_{at} or $l_{at:q}$ to agent q, then the act of reporting either or both of these values is said to be a subjectivity eliminated trust recommendation from $\boldsymbol{a}$ to q about t. Additionally, $l_{at:q} = fbvalue(c_{at}, d_{q,\psi})$ is said to be a subjectivity eliminated recommended trust value from $\boldsymbol{a}$ to q about t.*

We may think of $l_{uv:w}$ as the value l_{uv} transformed such that instead of being in reference to the disposition to trust of agent u, it is now in reference to the disposition to trust of agent w.

Instead of reporting l_{uv}, an agent u computes c_{uv} and communicates this percentile value to agent w. Given c_{uv}, agent w determines $l_{uv:w}$ and considers that as the recommended value. Agent u can also directly report $l_{uv:w}$ to agent w if $d_{w,\psi}$ is made available to agent u.

6.2.5 An Example of the Method in Use

As an example, consider $d_{Bob,\psi} = \langle 0.4, 0.4, 0.5, 0.6, 0.8, 0.8, 0.8, 0.8, 0.8, 0.9, 0.9 \rangle$ and $l_{Bob\ Carol} = 0.8$. Then $n_{Bob,\psi} = 11$ and $first(l_{Bob\ Carol}, d_{Bob,\psi}) = 5$. $c_{Bob\ Carol}$ is computed as follows:

$$
\begin{aligned}
c_{Bob\ Carol} &= percentile(l_{Bob\ Carol}, d_{Bob,\psi}) \\
&= \frac{100 \cdot first(l_{Bob\ Carol}, d_{Bob,\psi})}{n_{Bob,\psi} + 1} \\
&= \frac{100 \cdot 5}{11 + 1} = 41.67 percentile
\end{aligned}
$$

Now consider $d_{Alice,\psi} = \langle 0.2, 0.3, 0.3, 0.3, 0.5, 0.5, 0.5, 0.6, 0.8 \rangle$. Then:

$$\begin{aligned} l_{Bob\ Carol\ :\ Alice} &= fbvalue(c_{Bob\ Carol}, d_{Alice,\psi}) \\ &= d_{Alice,\psi,i} + f \cdot (d_{Alice,\psi,i+1} - d_{Alice,\psi,i}) \\ &= d_{Alice,\psi,4} + 0.17 \cdot (d_{Alice,\psi,5} - d_{Alice,\psi,4}) \\ &= 0.3 + 0.17 \cdot (0.5 - 0.3) = 0.33 \end{aligned}$$

where,

$$\begin{aligned} i &= \left\lfloor \frac{c_{Bob\ Carol} \cdot (n_{Alice,\psi} + 1)}{100} \right\rfloor \\ &= \left\lfloor \frac{41.67 \cdot (9 + 1)}{100} \right\rfloor = 4 \end{aligned}$$

and,

$$\begin{aligned} f &= \frac{c_{Bob\ Carol} \cdot (n_{Alice,\psi} + 1)}{100} - i \\ &= \frac{41.67 \cdot (9 + 1)}{100} - 4 = 0.17 \end{aligned}$$

Thus the subjectivity eliminated recommended trust from Bob to Alice about Carol is 0.33. This is in contrast to 0.8, which would have been the un-altered recommended trust. *Note:* Values have been rounded to two decimal places in the above example.

6.2.6 Experiment Design

Our objective is to test if the trust values recommended through the subjectivity-eliminated trust recommendation method are of higher quality than those given by the unmodified trust recommendation method in which trust values are conveyed without any alteration.

The quality of a recommended trust value is stated as its closeness to the trust value that the querying agent would assign to the target agent if it had direct experience with it.

The experiment takes a web of trust as input. We consider each edge (q, t) in the graph. Let's designate q as the querying agent and t as the target agent. We know that the trust of q in t is l_{qt}, which is the weight of the edge (q, t). The querying agent then receives trust recommendations (both un-altered – l_{at}, as well as subjectivity-eliminated – $l_{at:q}$) from each of the source agents of the target agent. In this scenario, not only do we have both the un-altered and the subjectivity-eliminated recommended trust values from each of the source agents but we also know what trust value the querying agent has assigned to the target agent based on direct experience. We therefore have a reference value with which we can compare the two recommended trust values.

If the value given by the subjectivity-eliminated trust recommendation method is significantly closer to the reference value than the one given by the unmodified trust recommendation method, we consider the experiment run as a success (hit) for our method. If the opposite is true, we consider it as a failure (miss). If both values are the same or are within a small range (0.03) of each other, we count a tie.

To facilitate the discussion we establish the following terminology:

α – recommended trust value given by the unmodified trust recommendation method which does not take subjectivity into account

β – recommended trust value derived from the subjectivity-eliminated trust recommendation method

γ – trust value depicting the source agent's trust in the target agent based on direct experience

Given G_ψ, a web of trust in context ψ, and min, the minimum number of elements in the dispositions to trust of the querying and source agents, the experiment is algorithmically described in Figure 6.1.

```
SUBJECTIVITY-EXPERIMENT(G_ψ, min)
 1  hits ← 0
 2  misses ← 0
 3  equals ← 0
 4  for each vertex q in G_ψ, such that n_{q,ψ} ≥ min
 5        do for each vertex t in G_ψ, such that edge (q,t) exists
 6              do γ ← l_{qt}
 7                 remove the edge (q,t)
 8                 for each vertex a in G_ψ, such that edge (a,t) exists,
                                 and n_{a,ψ} ≥ min
 9                       do α ← l_{at}
10                          β ← fbvalue(percentile(l_{at}, d_{a,ψ}), d_{q,ψ})
11                          if |α − β| < 0.03
12                             then equals++
13                          elseif |β − γ| < |α − γ|
14                             then hits++
15                          elseif |α − γ| < |β − γ|
16                             then misses++
17                 restore the edge (q,t)
18  print hits, misses, equals
```

Figure 6.1: Experiment design.

Given a large and diverse web of trust we can assume that there will be both hits and misses. However, if the number of hits is significantly larger than the

number of misses, we have an indication that the method is effective. On the contrary, if the number of misses is considerably greater than the number of hits or if there is no significant pattern then we may infer that the method is ineffective. The success rate of the method for a given experiment run is defined as: $success = \frac{hits}{hits+misses}$.

6.2.7 The Dataset for the Experiment

Consider the following environment:

There are N agents in the environment. The context of the trustworthiness of the agents is an action ψ. After an agent interacts with another agent, it quantifies its trust in that agent based on its experience. For example, after an agent Bob interacts with an agent Carol, he quantifies his trust in her based on the behavior that she exhibited. The scale for representing trust is of high resolution. An agent quantifies his trust subjectively according to his own disposition to trust. For example, an agent Alice would quantify her trust in Carol according to her own disposition to trust whereas Bob would do so according to his own. Each agent exhibits fairly consistent behavior in terms of action ψ. That is, if two agents Alice and Bob interact with an agent Carol, then there is high probability that Carol will exhibit the same behavior towards both in terms of action ψ.

This is clearly an environment where subjectivity would be an issue in trust recommendations. Arguments: 1) Each agent quantifies its trust subjectively according to its own disposition to trust. 2) The scale for representing trust is of high resolution. 3) The behavior exhibited by an agent is fairly consistent, thus source agents assign different trust values not due to different experiences, but due to differences in disposition to trust.

We generate a simulated web of trust representing the above described environment. A simulated dataset is used since we have not come across any publicly available real and large web of trusts which employ a high resolution scale for trust representation. The Advogato web of trust that we have used in our other experiments employs a scale comprising of only four values.

The simulated web of trust is generated as described in Figure 6.2. n is the number of vertices of the graph, m is the number of edges of the graph, and G is the generated graph.

The trustworthiness factor $\underline{tf}_u$ represents the experience that other agents would have as a result of their interaction with an agent u. Since $\underline{tf}_u$ remains constant for agent u, any agent that interacts with it has the same experience. Trustworthiness factor $\in [0, 1]$.

The skew factor $\underline{af}_u$ represents the individual disposition to trust of an agent u. Although different agents have the same experience with a given agent, they each assign it a different trust value based on their own disposition to trust. Skew factor $\in [0, 2]$.

The weight of an edge (u, v), which is also the trust value l_{uv}, is obtained as $\underline{tf}_v^{\underline{af}_u}$. If the skew factor $\underline{af}_u$ is less than 1, the trustworthiness factor $\underline{tf}_v$

GENERATE-WEB-OF-TRUST(n, m)

1 generate a directed weighted random graph (without multiple edges between any two vertices), $G(V, E, [0, 1])$, where V is the set of vertices and E is the set of edges, $|V| = n$, and $|E| = m$
2 label the vertices as $u_1 \dots u_n$
3 with each vertex u_i, associate a trustworthiness factor $\underline{\underline{tf}}_{u_i}$, randomly (uniform) selected from the interval $[0, 1]$, $i = \{1 \dots n\}$
4 with each vertex u_i, associate a skew factor $\underline{\underline{af}}_{u_i}$, randomly (uniform) selected from the interval $[0, 2]$, $i = \{1 \dots n\}$
5 **for** each vertex u_i, $i = \{1 \dots n\}$
6 **do** let $\{v_1 \dots v_{\eta_i}\}$ be the set of vertices who have an incoming edge from u_i, where η_i is the number of outgoing edges of u_i
7 **for** each vertex v_j, $j = \{1 \dots \eta_i\}$
8 **do** assign the weight $\underline{\underline{tf}}_{v_j}^{\underline{\underline{af}}_{u_i}}$ to the edge (u_i, v_j)
9 **return** G

Figure 6.2: Pseudo code for generating the web of trust.

is skewed upwards. Otherwise, if the skew factor $\underline{\underline{af}}_u$ is greater than 1, $\underline{\underline{tf}}_v$ is skewed downwards. The resulting weight $\in [0, 1]$.

The trust values lie in the set of real numbers between 0 and 1. The resolution for expressing trust is therefore high.

6.2.8 Experiment Runs and Results

The experiment and the generation of the dataset have been implemented in Java with the Java Graph library (JGraphT).

We iterate m from 500 through 6000 with an interval of 500. For each iteration of m, we generate three datasets, with the number of vertices $n = 100$. The mean outgoing edges per vertex for each iteration is given as m/n. For each of the datasets, we run the experiment and obtain the number of *hits*, *misses* and *equals*. The results of the experiment runs are given in Table 6.1.

6.2.9 Discussion of Experiment Results

The first observation we can make is that in each of the experiment runs the number of *hits* is significantly higher than the number of *misses*, thus resulting in a high success rate for the subjectivity-eliminated trust recommendation method.

The second observation is that the success rate of the method rises as the average number of outgoing edges per vertex increases. We observe that at $m/n = 60$, the success rate for each of the three experiment runs is a high 91%.

Table 6.1: Experiment runs, $n = 100, min = 1$.

m	m/n	Dataset 1				Dataset 2				Dataset 3				***success***
		hits	*misses*	*equals*	*success*	*hits*	*misses*	*equals*	*success*	*hits*	*misses*	*equals*	*success*	**(mean)**
500	**5**	1485	585	312	72%	1416	637	395	69%	1552	570	328	73%	**71%**
1000	**10**	6186	2156	1400	74%	5989	2291	1586	72%	5881	2230	1507	73%	**73%**
1500	**15**	15354	3447	3113	82%	13800	4379	3289	76%	13671	4679	3336	75%	**78%**
2000	**20**	26685	6398	5635	81%	26725	6745	5436	80%	26039	6707	5982	80%	**80%**
2500	**25**	41075	9721	9494	81%	39850	10475	9935	79%	41967	8784	9439	83%	**81%**
3000	**30**	62377	11125	13950	85%	56182	13836	16906	80%	58813	13429	15070	81%	**82%**
3500	**35**	91172	12075	16157	88%	84481	14510	20231	85%	85920	14055	18683	86%	**86%**
4000	**40**	115889	15971	23770	88%	111402	17523	26743	86%	106994	21135	27323	84%	**86%**
4500	**45**	138885	23663	34172	85%	148721	16879	31094	90%	141431	20161	34630	88%	**88%**
5000	**50**	171168	23617	46827	88%	173932	25857	42311	87%	179835	22475	39458	89%	**88%**
5500	**55**	204641	29591	58408	87%	218158	25927	49689	89%	220398	25487	46981	90%	**89%**
6000	**60**	263296	24550	60324	91%	269041	25055	55684	91%	255199	26263	67688	91%	**91%**

Even with lower numbers of outgoing edges, the method still outperforms the one that does not account for subjectivity.

These results provide a positive indication that the subjectivity-eliminated trust recommendation method is more effective than the unmodified method. However, we must note that at this moment these results hold true for the type of web of trust used for the experiments. The primary characteristics of this type of web of trust include: wired as a directed random graph, agents exhibit consistent behavior and disposition to trust, both factors are uniformly distributed, a trust value is a function of the behavior of the trustee and the disposition to trust of the truster, and the trust values lie in the interval $[0, 1]$.

Although the simulated web of trust takes into account real world issues such as truster's bias in forming trust beliefs, some other aspects are simplified. For example, it is wired as a random graph. Another simplification is the uniform distribution of the dispositions to trust and the trustworthiness of the agents. These simplifications may very well not have any impact on the effectiveness of the method however that is a hypothesis which would need to be tested.

An evident direction for future work is to test the method on a real web of trust or a closely approximated simulated web of trust. Some ideas for generating a more realistic web of trust include: 1) connectivity based on small-world [138] or scale-free networks [7], which are better representations of social networks, and 2) representing the dispositions to trust and the trustworthiness of the agents by a distribution such as normal or power-law.

6.2.10 A Limitation of the Proposed Method

We have proposed to represent an agent's disposition to trust by the collection of the trust values that he has assigned. This proposal is based on the idea that if an agent has a pattern of assigning high values of trust, we may infer that the agent has a high disposition to trust, and vice versa.

However, it is possible that an agent (such as Bob in our running example) assigns mostly high values of trust because he encounters agents with mostly high trustworthiness (and not because Bob has a high disposition to trust). Similarly, it is possible that an agent (such as Alice) assigns a majority of low values of trust because she encounters a large proportion of agents with low trustworthiness. In such cases, the collection of the trust values that an agent has assigned would not be an accurate representation of his or her disposition to trust.

Let's consider an agent who encounters and evaluates agents that are uniformly distributed over A (the set of all agents). Then the set of agents evaluated (given as the set $O_{u,\psi}$ for an agent u in the context ψ in definition 34) is likely to include agents that span the spectrum of trustworthiness. This is the case in our simulated data set for all agents.

Our method for the elimination of subjectivity would clearly misinterpret the trustworthiness of an agent if the disposition to trust of either the querying or the source agent is misrepresented. For example, if an agent only evaluates other agents from a specific cross-section of trustworthiness. We propose the following alternatives to overcome this limitation:

1. Assess the disposition to trust of the agents through a trust scale (such as Rotter's interpersonal trust scale) as discussed in Section 6.2.2. However, the disadvantage of this approach is that it demands input from the individual.

   Alternatively, we may require agents to rate their own disposition to trust manually. We followed this approach in [118] (a paper co-authored with Saadi, Pierson, and Brunie), where administrators of a domain collectively select a common disposition to trust from the range $[0, 9]$. The trustworthiness of remote sites is then evaluated with respect to the selected disposition to trust instead of the varying individual dispositions to trust of the administrators.

2. Compute the intersection between the sets $O_{q,\psi}$ and $O_{a,\psi}$, where q is the querying agent and $\boldsymbol{a}$ is the source agent. Then consider only those agents for the disposition to trust of the two agents that lie in the intersection set. Having evaluated the same set of agents would make the dispositions of trust of the querying agent and the source agent comparable.

   However, this approach may not always work since the intersection set may turn out to be empty. To alleviate this issue, we may require each agent to evaluate a certain global set of agents before they utilize the elimination of subjectivity method. Thus any two given agents would have at least that set of agents in common.

6.3 An Application of Trust Propagation: Access Control in Multi-Domain Environments

In this part of the chapter, we focus on one of the important applications of trust propagation, namely access control in multi-domain environments. The goal of access control in multi-domain environments is to provide users with seamless access to resources at their home site as well as at foreign sites. Access control for such an environment is inherently challenging since sites may have no knowledge of remote users who visit them. Traditional access control models based on roles (as in RBAC [48]) or identities (as in IBAC [71]) are often not suitable for such environments since roles or identities that exist at one site may not exist at another.

We present a solution to access control in multi-domain environments based on trust propagation. The proposal is to consider the trustworthiness level of users as the criterion for granting access to resources. A site has direct knowledge of the trustworthiness of its own users. Whereas, the trustworthiness of an unknown user can be determined through trust propagation. Trust propagation enables the foreign site to acquire trust in the unknown user through a path of trust recommendations that link the site to the user. For example, a site x may acquire trust in an unknown user u, if u's home site y which is trusted by x, makes a recommendation to x about u.

A number of works, which include [92, 5, 73, 78, 119, 124, 113], have introduced solutions based on trust propagation to access control in multi-domain environments. Our proposal is similar to Saadi et al. [119], Richardson et al. [113], and some other models in the sense that we use the multiplication operator to implement the trust propagation function.

Compound propagated trust is computed by iteratively multiplying the trust values on the path from a querying entity to the target entity. The novel contribution of our work is experimental evidence that there is a positive correlation between trust acquired through direct interaction and trust acquired through trust propagation using the iterative multiplication strategy.

The data set used for the experiment is the real web of trust of Advogato.org. The instance of the web of trust of Advogato.org used for the experiment comprises of over $11,000$ vertices (users) and over $50,000$ directed weighted edges (trust relationships between users). Results show that a substantial positive linear correlation exists between trust values established from direct experience and propagated trust values derived through the iterative multiplication approach. To the best of our knowledge, this is the first work to provide evidence of this correlation based on a real and large web of trust.

6.3.1 Problem Setting

The environment comprises of $\underline{nx}$ number of sites given as the set $X = \{x_1, x_2, \ldots, x_{\underline{nx}}\}$. A site is characterized by its association to a set of member users, a set of resources under its ownership, and an access control

policy. Some examples of sites include university campuses, corporate offices, airports, etc.

The set of member users associated with a site x is given as $U_x = \{u_{x,1}, u_{x,2}, \ldots, u_{x,\underline{nu}_x}\}$, where $\underline{nu}_x$ is the number of member users. For simplicity we assume that the set of users of any two sites x and y are disjoint, that is $U_x \cap U_y = \phi$.

The set of resources of site x is given as $R_x = \{r_{x,1}, r_{x,2}, \ldots, r_{x,\underline{nr}_x}\}$, where $\underline{nr}_x$ is the number of resources of site x.

A user may request access to a resource at his home site or he may roam in the environment and request access to the resources of foreign sites. Each site has an access control policy that determines if a user is qualified to access a resource that he has requested.

The goal is to make the access control process for a user as ubiquitous at a foreign site as it is at his home site.

6.3.2 The Access Control Model

In this section we present our model for access control in multi-domain environments that uses the iterative multiplication strategy for trust propagation. The model is founded upon our extended general framework presented in Section 6.1.

Definition 39. ***User.*** *A user u is defined as an agent such that it has no relationship of the form $u\ T\ a$, where $\boldsymbol{a}$ is any agent in A (the set of all agents in the environment). That is, $(u, a) \notin T, \forall a \in A$.*

A user is defined as an agent who does not form trust relationships towards other agents. The sole interest of a user in this model is to access desired resources. The trust relationships are formed among sites and by sites towards their member users.

Definition 40. ***Resource.*** *A resource r is defined as any non-agent entity or service of potential value to users.*

Definition 41. ***Site.*** *A site x is defined as an agent such that there exists a set $U_x = \{u_{x,1}, u_{x,2}, \ldots, u_{x,\underline{nu}_x}\}$, where $u_{x,i}$ is a user and trust of x in $u_{x,i}$ exists in a context ψ for $i \in \{1 \ldots \underline{nu}_x\}$. For any two sites x and y, $U_x \cap U_y = \phi$. Moreover, there exists a set $R_x = \{r_{x,1}, r_{x,2}, \ldots, r_{x,\underline{nr}_x}\}$, where $r_{x,i}$ is a resource. For any two sites x and y, $R_x \cap R_y = \phi$. Additionally, the set of trustees of site x may only include members of set U_x and other sites. That is, $\forall (x, a) \in T, a \in U_x$ or $\boldsymbol{a}$ is a site.*

A site is defined as an agent who has a set of users associated with it and it has trust in each of those users. The set of a site's users is mutually exclusive of the sets of users of other sites. Moreover, a site is also associated with a set of resources, which is also mutually exclusive of the sets of resources of other sites. A site has trust relationships only towards its member users or other sites.

Definition 42. ***Site-to-User Trust Path.*** *A site-to-user trust path $p_\psi(x_1, u)$ is defined as a trust path such that the sequence of agents $\langle x_1 \dots u \rangle$ is composed of n sites $x_1 \dots x_n$, followed by a user u, where $n \geq 2$. In other words, $p_\psi(x_1, u) = \langle x_1 \dots x_n, u \rangle$, where $x_1 \dots x_n$ are sites and u is a user.*

Definition 43. ***Function ptrust'.*** *$ptrust'(p_\psi(x_1, u))$ is a realization of the function ptrust (Definition 32). $ptrust'$: site-to-user trust path $\rightarrow [0, 1]$. $ptrust'$ is implemented as follows:*

$$\begin{aligned} & ptrust'(p_\psi(x_1, u)) \\ = \; & ptrust'(\langle x_1 \dots x_n, u \rangle) \\ = \; & l_{x_1 x_2} \times \dots \times l_{x_{n-1} x_n} \times l_{x_n u} \\ = \; & (\prod_{i=1}^{n-1} l_{x_i x_{i+1}}) \times l_{x_n u} \end{aligned}$$

Please see the next subsection for our reasoning for using multiplication.

Definition 44. ***Access Control Policy.*** *An access control policy of a site x is defined as a set $\underline{ACP}_x$ of pairs of the form (r, min_r), where $r \in R_x$ and $min_r \in [0, 1]$. A one-to-one correspondence exits between the set R_x and the set $\underline{ACP}_x$. Site x grants a user u access to a resource r only if $l_{xu} \geq min_r$ or $ptrust'(p_\psi(x, u)) \geq min_r$.*

With each resource r, the site x defines a threshold value min_r. The access control policy of a site lists all its resources and associated thresholds.

Access is granted to a user u that requests a resource r at a site x only if $l_{xu} \geq min_r$ or $ptrust'(p_\psi(x, u)) \geq min_r$. In other words, access to a resource is granted if the site has equal or greater trust in the requesting user than the threshold for that resource.

The access control policy defined in definition 44 can be classified as Mandatory Access Control (MAC) [89], since access is controlled based on mandatory rules determined by a central site. However, it is important to note that in our model, the user u may or may not be a member of the site x. If u is a member of the site x then the site has direct knowledge of the user's trustworthiness. In case u is not a member then access may still be granted if trust in u can be established through trust propagation and $ptrust'(p_\psi(x, u))$ passes the trustworthiness threshold.

What makes the model ubiquitous is that a site does not need to have pre-defined access rights for a certain user to be able to grant them access to resources. The site can establish trust in a previously unknown user through trust propagation and it can grant them access based on that acquired trust. From the user's point of view, access to resources at foreign sites is as seamless as at their home site.

The access control model that we have described, only allows a simple scalar threshold as the condition for access to a particular resource. A more sophisticated language for defining access control conditions would be required for a

practical deployment of the model. A few things that the language should handle include: 1) different thresholds for different operations such as read, write, and execute, and 2) specifying contextual conditions such as those based on time, location etc.

Reasoning for Using Multiplication

The suggested propagated trust value is the product of all the trust values on the path. We implement the function as such for its simplicity and intuitiveness. We consider a few examples to illustrate our point.

Let's assume that all the trust values on the path are 1. The trust value suggested by the function in this case would be 1, which reflects the fact that absolute trust exists throughout the chain.

As another case let's consider that any one or more of the trust values on a path are 0. That is, one of the sites has no trust in the entity that it has a trust relationship with. The trust value suggested by the function would be 0. Thus the fact that one of the sites does not trust an entity on the path is appropriately reflected in the suggested value.

Let's now consider a path of length 3 with each of the trust values as 0.9. The suggested trust value would be $0.9 \times 0.9 \times 0.9 = 0.73$. Although each of the sites has a high trust of 0.9 in the recommended site or user, the suggested trust value is a lower 0.73. This value is reflective of the degree of separation between the source site and the target user. Intuitively, trust attenuates as the degree of separation between the source site and the target user grows.

As the final example we consider the path $\langle x_1, x_2, x_3, u\rangle$ with $t(x_1, x_2) = 0.1$, $t(x_2, x_3) = 0.8$, and $t(x_3, u) = 0.9$. The suggested trust value would be $0.1 \times 0.8 \times 0.9 = 0.07$. Although x_2 and x_3 have very high trust in x_3 and u respectively, since x_1 has low trust in x_2, the propagated trust value remains low.

We refer the reader to [135] (section 20.2.2) and [113] (section 3) for further discussion on using multiplication as the operator for trust propagation.

6.3.3 The Experiment

The objective of this experiment is to determine whether it is prudent to establish trust in an unknown entity based on trust propagation. More precisely, whether a substantial positive correlation exists between direct trust and propagated trust. Direct trust is the amount of trust that a source agent establishes in a target agent based on direct experience.

Tools and Techniques: Correlation

Correlation is a coefficient that measures the strength and the direction of the linear relationship between two variables.

The correlation coefficient lies on the interval $[-1, 1]$. Values near 1 and -1 indicate a strong linear relationship between the two variables. Values close to

0 indicate a weak relationship. A positive value implies that the relationship is proportional, that is, increase in the value of one variable is likely to result in the increase of the value of the other variable. A negative value implies an inverse relationship.

The correlation r between two variables x and y is given as follows:

$$r = \frac{1}{n-1} \sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{s_x} \right) \left(\frac{y_i - \bar{y}}{s_y} \right) \tag{6.1}$$

where, $\bar{x}$ and s_x are the *mean* and the *standard deviation* of variable x, and n is the size of the bivariate data.

Correlation computed with this specific method is also known as the Pearson Product-Moment Correlation.

Experiment Design

In a web of trust, the weight of an edge from a truster to a trustee represents the direct trust that the truster holds for the latter. If an alternate path exists from that truster to the trustee, we can compute the propagated trust between the two vertices from that path. Based on these observations we design the experiment as follows:

We consider every edge in a given web of trust. An exception is those edges that have the same source and target vertex. The direct trust of an edge's source vertex (the truster) in its target vertex (the trustee) is the weight of that edge. Having noted the direct trust from the truster to the trustee, we consider the scenario that direct trust between the two vertices does not exist. We remove the direct edge and using Dijkstra's algorithm, we find an alternate path from the truster to the trustee. If an alternate path exists, we obtain the propagated trust using the *ptrust'* function. Now we know the direct trust of the truster in the trustee as well as the propagated trust. After obtaining all such pairs of direct trust and propagated trust, we calculate the correlation between the two variables. It is important to note that the values of direct trust and propagated trust are obtained independently of each other in this experiment.

The Dijkstra's algorithm may return several alternate shortest paths. In this experiment, we always consider the first path that is returned by the algorithm. As future work, a variation on the experiment could be to identify and select the path that yields the optimal trust value. Such a path may or may not be the shortest one.

The experiment is algorithmically described in Figure 6.3. G is a web of trust. $dijkstra(x, y)$ is a function which returns a path from vertex x to vertex y, given as $p(x, y)$, using Dijkstra's shortest-path algorithm. $correlation(\overrightarrow{\mathcal{X}}, \overrightarrow{\mathcal{Y}})$ is a function which returns the correlation r between two variables represented by vectors $\overrightarrow{\mathcal{X}}$ and $\overrightarrow{\mathcal{Y}}$.

The experiment has been implemented in Java with the Java Graph library (JGraphT).

```
TRUST-PROPAGATION-EXPERIMENT(G)
 1  i ← 0
 2  for all edges in G, whose source vertex (given as a_s) and
                target vertex (given as a_t) are not the same
 3        do direct-trust ← t(a_s, a_t)
 4           remove the edge (a_s, a_t)
 5           p(a_s, a_t) ← dijkstra(a_s, a_t)
 6           if |p(a_s, a_t)| > 0
 7              then prop-trust ← ptrust'(p(a_s, a_t))
 8                   add direct-trust to vector D⃗ at index i
 9                   add prop-trust to vector P⃗ at index i
10                   i ← i +1
11           restore the edge (a_s, a_t)
12  r ← correlation(D⃗, P⃗)
13  print r
```

Figure 6.3: Experiment Design.

The Data Set for the Experiment

The data set that we use for our experiment is the real web of trust of Advogato.org [87, 88]. The reader is referred to Section 4.8.1 for a detailed discussion of Advogato.org. The instance of the Advogato web of trust referenced in this experiment was retrieved on November 19, 2007. The choice of trust values in this instance are *master*, *journeyer* and *apprentice*, with *master* being the highest level in that order. The web of trust comprises of $11,558$ users and $51,119$ trust ratings. The distribution of trust values is as follows: *master*: $17,478$, *journeyer*: $22,894$, and *apprentice*: $10,747$. To conform this web of trust to our framework, we substitute its three trust values as follows: $master = 1.0$, $journeyer = 0.66$, and $apprentice = 0.33$.

6.3.4 Experiment Runs, Results and Analysis

We run the experiment with the adapted Advogato web of trust as G. The number of instances when an alternate path was found between two vertices with a direct edge is $44,959$. The final value of i in the algorithm of the experiment gives this value. A histogram of the lengths of the alternate paths is given in Figure 6.4. The outcome of the experiment, the correlation between direct trust and propagated trust, is 0.61 (rounded down to two decimal places).

The histogram shows that the edge count is at most 3 for over 96% of the instances when a path is found from the source vertex to the target vertex. As we discussed in Section 6.3.2, fewer edges on the path between two entities leads to lower attenuation of trust propagated over that path. The observation

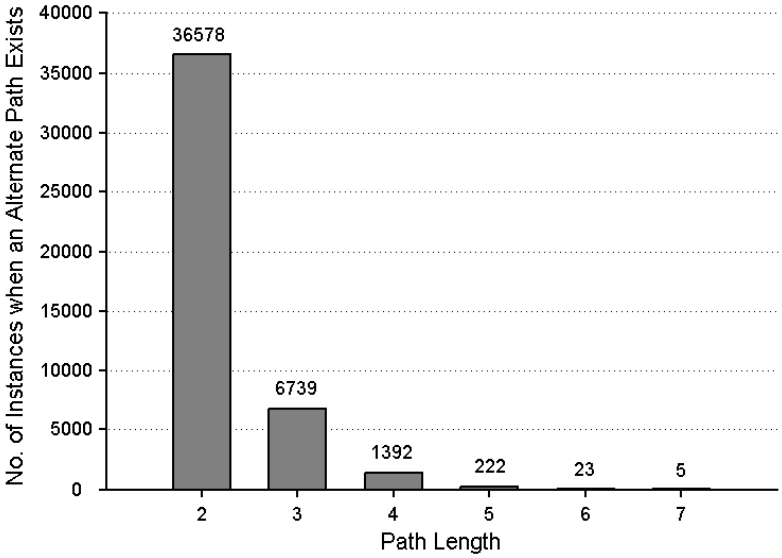

Figure 6.4: Histogram of Path Lengths.

thus implies that a high percentage of the propagated trust values have low attenuation.

The experiment indicates that a substantial positive linear correlation (0.61) exists between direct trust and propagated trust acquired through the iterative multiplication approach. We note again that the values of direct trust and propagated trust are obtained independently of each other in the experiment.

6.4 Privacy

The ideas of trust recommendation and privacy are hard to reconcile since by definition trust recommendation is the disclosure of a feedback value to another agent. However, we can imagine approximations of the trust recommendation technique that are better at preserving the privacy of the source agent.

One such technique is as follows: Instead of reporting the feedback value, the source agent may only respond to a query that demands whether his trust in the target agent is higher (or equal to) or lower than a given level. For example, Alice could ask Bob whether his trust in Carol is higher (or equal to) or lower than 0.8. A binary response of *higher* or *lower* may be sufficient for Alice to make a decision whether to trust Carol or not. The disadvantages of this technique are that: 1) the privacy of the source agent is not completely preserved, that is, the querying agent gains partial information about the feedback value; 2) an adversary could make repeated queries in order to narrow down on the feedback value. This attack may be prevented if the source agent is able to identify repeated queries by the adversary (and any members in his clique).

Another technique is to perturb the feedback value before providing it to the querying agent. This technique also has the potential to divulge partial information about the feedback value. An attacker could also sabotage this technique by repeated queries. If the value is perturbed randomly each time, then the attacker can use distribution reconstruction to derive the feedback value. However, this may be prevented if the perturbation is kept constant for a feedback value.

6.5 Summary

In this chapter we studied some issues related to trust recommendation and propagation. The techniques of trust recommendation and propagation are related to reputation as they also enable agents to establish trust in other agents. In Section 6.1, we extended our general framework to accommodate trust recommendation and propagation.

In Section 6.2, we explored the problem of subjectivity in trust recommendation. We argued that subjectivity prevents the real meaning of a trust value from being conveyed by one agent to another. A method that attempts to eliminate subjectivity from trust recommendation was then presented. The method proposes to use percentiles, which are equally meaningful among two agents.

Experiments conducted on simulated web of trusts indicate that the method is fairly effective for the elimination of subjectivity from trust recommendation. The method is non-intrusive and does not require any change in how agents locally evaluate other agents. Furthermore, the method does not involve any third party mediation, thus making it suitable for decentralized networks. Validation of the experiment results on a real web of trust or a closely approximated simulated web of trust is suggested for establishing further confidence in the proposed method.

In Section 6.3, we analyzed the iterative multiplication strategy for trust propagation employed by various access control models for multi-domain environments. Through an experiment on the real and large web of trust of Advogato.org, we showed that a substantial positive linear correlation exists between direct trust and propagated trust acquired through the iterative multiplication approach. This result raises confidence in the notion of establishing trust in an unknown entity through the discussed trust propagation method.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this thesis, we focused on privacy preserving reputation systems. These systems compute reputation scores without revealing the individual feedback of any user. Preserving the privacy of users gives them the freedom to provide truthful feedback. Our objective was to construct privacy preserving reputation protocols that are decentralized, do not require specialized platforms nor trusted third parties, and protect privacy under standard adversarial models. Another primary goal was to design the protocols such that they are more efficient than the existing protocols that meet the above criteria.

We presented Round-Trip, a privacy preserving protocol for computing reputation under the semi-honest adversarial model. The protocol draws its strength from elements that include secure multi-party computation, data perturbation, seed agents, and most importantly, trust awareness in the context of preserving privacy. Trust awareness allows a feedback provider to select trustworthy agents, whom he can rely on to preserve his privacy. It also enables the feedback provider to quantify the probability that his privacy will be preserved. Since the risk to privacy can be quantified, a feedback provider can decide to abstain from providing feedback if the risk is too high. The ability to abstain means that agents can achieve up to perfect privacy, which was previously considered impossible in decentralized additive reputation systems.

Experiments conducted on the real and large web of trust of Advogato.org demonstrate that the protocol is able to guarantee the privacy of around 40% of the feedback providers. Although this success rate is low, we observe from the experiments that even if the remaining agents abstain, the protocol still computes very accurate reputation scores as the mean of the available feedback. The protocol requires $O(n)$ messages to be exchanged, where n is the number of feedback providers in the protocol.

We also presented the k-Shares privacy preserving reputation protocol. The k-Shares protocol is secure under the semi-honest model. In contrast to the Round-Trip protocol, it preserves the privacy of a high majority of feedback providers. The protocol builds upon the techniques of secret sharing and trust awareness. k is a constant that represents the maximum number of trustworthy agents that a feedback provider can rely on to preserve his privacy, where $k \ll n$, and n is the number of feedback providers in the protocol. Keeping k significantly less than n results in linear message complexity for the protocol, that is, $O(n)$.

Experiments conducted on the web of trust of Advogato.org show that the privacy of around 85% of the feedback providers can be ensured with k as small as 2. The experiments also demonstrate that increasing k to values approaching n gives little improvement in this percentage. Thus the k-Shares protocol achieves near optimal success rate while maintaining linear message complexity. Agents in the k-Shares protocol can also quantify the risk to their privacy due to trust awareness. Therefore, the protocol may be extended such that agents can abstain if the risk to their privacy is not acceptable.

Our protocol for the disruptive malicious model is a version of our k-Shares protocol enhanced with cryptographic constructs, which include an additive homomorphic cryptosystem and zero knowledge proofs. The number of messages exchanged is $O(n) + O(log\ N)$, where n is the number of feedback providers in the protocol and N is the total number of agents in the system.

All three protocols are fully decentralized and do not require any specialized hardware nor trusted third parties. Each of the protocols has lower communication complexity than the existing protocols that meet these criteria and that are secure under the same adversarial model.

In this thesis, we also addressed the problem of subjectivity in trust recommendation. Subjectivity prevents the real meaning of a trust value from being conveyed by one agent to another. We presented a method for the elimination of subjectivity from trust recommendation, which proposes to exchange percentiles rather than raw trust scores. Experiments conducted on a simulated web of trust demonstrate that the method gives more accurate trust recommendations than the unmodified method, up to 91% of the time. The method is non-intrusive and does not require any change in how agents locally evaluate other agents. Furthermore, the method does not involve any third party mediation, thus making it suitable for decentralized networks.

Another contribution was the analysis of the iterative multiplication strategy for trust propagation, which is employed by a number of access control models for multi-domain environments. Through an experiment on the web of trust of Advogato.org, we showed that a substantial positive linear correlation (0.61) exists between direct trust and propagated trust acquired through the iterative multiplication approach. This result raises confidence in the notion of establishing trust in an unknown entity through trust propagation based on iterative multiplication.

7.2 Future Work

As future work, we would like to address the issue of privacy in conjunction with other challenges faced by reputation systems. As discussed in Section 2.2.3, these challenges include sybil attack, self-promotion, slandering, whitewashing, and oscillation. Existing solutions to these problems are oriented mainly towards non-privacy preserving reputation systems. Let's consider the problem of detecting sybil attacks in reputation systems. Advogato.org accomplishes this task by running a network flow operation on a global view of the trust graph. This approach cannot be directly applied to decentralized privacy preserving reputation systems, because a global view of the trust graph is not available, and moreover the feedback of agents must remain private. We would like to explore the use of privacy enhancing technologies (such as secure multi-party computation, data obfuscation, etc.) for adapting existing approaches to decentralized privacy preserving reputation systems.

Another area of interest is privacy preserving reputation systems for location sensitive networks. Consider a network where feedback is generally assigned to immediate neighbors. Examples of such networks include MANETs and VANETs. If an agent u assigns feedback to some agent v at location x, then it can be inferred that agent u was also present at location x. This is clearly a breach of agent u's privacy. The challenge is to develop a protocol that allows agents to assign feedback without compromising their geographical location. We would like to explore a solution based on the k-anonymity model [130].

Work in progress includes a new privacy preserving reputation protocol that builds upon the data perturbation technique and the Expectation-Maximization (EM) algorithm for distribution reconstruction [3]. Each i^{th} source agent sends the value $z_i = x_i + y_i$ to the querying agent, where x_i is its feedback value and y_i is a random number large enough to hide x_i. Given $z_1 \dots z_n$ and the distribution of $y_1 \dots y_n$, the querying agent uses the EM algorithm to reconstruct the distribution of $x_1 \dots x_n$. The querying agent can subsequently use the reconstructed distribution of the feedback values to derive the reputation score. The advantage of this protocol is that it is very efficient in terms of the number and size of messages exchanged. Moreover, an agent does not need to rely on fellow agents for its privacy. However, preliminary results show that the protocol is suitable only for computing the reputation of target agents who have a large number of source agents (due to the effect of the size of the dataset on the success of the EM algorithm). We are also interested in developing a version of the EM-based protocol that is secure under the malicious model. An incorrect distribution of the y values can result in failure of the reconstruction algorithm. Therefore, an issue to be addressed under the malicious model is to ensure that each i^{th} source agent generates the random number y_i in the distribution that is expected by the querying agent.

Bibliography

[1] A. Abdul-Rahman and S. Hailes. Supporting trust in virtual communities. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*, January 2000.

[2] C. Adams and S. Farrell. Internet x.509 public key infrastructure: Certificate management protocols. RFC 2510, March 1999.

[3] D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *Proc. of the 20th ACM Symposium on Principles of Database Systems*, 2001.

[4] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, Dallas, Texas, 2000.

[5] S. I. Ahamed, M. M. Haque, and N. Talukder. Service sharing with trust in pervasive environment: Now its time to break the jinx. In *Proc. of the 23rd Annual ACM Symposium on Applied Computing (SAC 2008)*, Fortaleza, Cear, Brazil, March 2008.

[6] Aladdin Knowledge Systems Ltd. Attack intelligence research center annual threat report – 2008 overview and 2009 predictions. http://www.aladdin.com/pdf/airc/AIRC-Annual-Threat-Report2008.pdf, 2008.

[7] R. Albert and A.-L. Barabasi. Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74, January 2002.

[8] A. Amirbekyan and V. Estivill-Castro. A new efficient privacy-preserving scalar product protocol. In *Proceedings of the Sixth Australasian Conference on Data Mining and Analytics*, 2007.

[9] E. Androulaki, S. G. Choi, S. M. Bellovin, and T. Malkin. Reputation systems for anonymous networks. In *Proceedings of the 8th Privacy Enhancing Technologies Symposium (PETS 2008)*, 2008.

[10] R. Aringhieri, E. Damiani, S. D. C. D. Vimercati, S. Paraboschi, and P. Samarati. Fuzzy techniques for trust and reputation management in

anonymous peer-to-peer systems. *Journal of the American Society for Information Science and Technology*, 57(4):528537, February 2006.

[11] M. J. Atallah and W. Du. Secure multi-party computational geometry. In *Proceedings of the Seventh International Workshop on Algorithms and Data Structures (WADS 2001)*, 2001.

[12] R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984.

[13] O. Baudron, P.-A. Fouque, D. Pointcheval, J. Stern, and G. Poupard. Practical multi-candidate election system. In *Proceedings of the Twentieth Annual ACM Symposium on Principles of Distributed Computing*, 2001.

[14] M. Belenkiy, M. Chase, C. C. Erway, J. Jannotti, A. Kupcu, A. Lysyanskaya, and E. Rachlin. Making p2p accountable without losing privacy. In *Proceedings of the 2007 ACM Workshop on Privacy in Electronic Society*, 2007.

[15] M. Belenkiy, M. Chase, M. Kohlweiss, and A. Lysyanskaya. P-signatures and noninteractive anonymous credentials. In *Theory of Cryptography*, 2008.

[16] E. Bertino, E. Ferrari, and A. C. Squicciarini. Trust-x: A peer-to-peer framework for trust establishment. *IEEE Transactions on Knowledge and Data Engineering*, 16(7):827 – 842, July 2004.

[17] G. A. Bigley and J. L. Pearce. Straining for shared meaning in organization science: Problems of trust and distrust. *Acad. Management Rev.*, 23(3):405421, 1998.

[18] Y. Bo, Z. Min, and L. Guohuan. A reputation system with privacy and incentive. In *Proceedings of the Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD'07)*, 2007.

[19] S. D. Boon and J. G. Holmes. *Cooperation and Prosocial Behaviour*, chapter The Dynamics of Interpersonal Trust: Resolving Uncertainty in the Face of Risk, pages 190 – 211. Cambridge University Press, 1991.

[20] S. Buchegger and J.-Y. L. Boudec. Performance analysis of the confidant protocol (cooperation of nodes: Fairness in dynamic ad-hoc networks). In *Proceedings of the Third ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC'02)*, Lausanne, Switzerland, June 2002.

[21] S. Buchegger and J.-Y. L. Boudec. A robust reputation system for peer-to-peer and mobile ad-hoc networks. In *Proceedings of P2PEcon 2004*, Harvard University, Cambridge, MA, USA, June 2004.

[22] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *EUROCRYPT 2001*, 2001.

[23] J. Camenisch, A. Lysyanskaya, and M. Meyerovich. Endorsed e-cash. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2007.

[24] R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

[25] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings of the 42nd Annual IEEE Symposium on the Foundations of Computer Science (FOCS'01)*, 2001.

[26] L. Capra. Engineering human trust in mobile system collaborations. In *Proceedings of the 12th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, Newport Beach, CA, USA, 2004.

[27] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):8488, 1981.

[28] D. Chaum. Blind signatures for untraceable payments. In *Proc. Advances in Cryptology (CRYPTO '82)*, 1982.

[29] D. Chaum. Blind signature systems. In *Advances in Cryptology (CRYPTO'83)*, 1983.

[30] R. Christie and F. L. Geis. *Studies in Machiavellianism*. Academic Press, New York, 1970.

[31] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu. Tools for privacy preserving distributed data mining. *SIGKDD Explorations*, 4(2):28–34, January 2003.

[32] C. Costa and J. Almeida. Reputation systems for fighting pollution in peer-to-peer file sharing systems. In *Proceedings of the Seventh IEEE International Conference on Peer-to-Peer Computing*, October 2007.

[33] CyberSource Corporation. Cybersource eleventh annual online fraud report. http://www.cybersource.com/, 2010.

[34] I. Damgard and M. Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In *Proceedings of the 4th International Workshop on Practice and Theory in Public Key Cryptography: Public Key Cryptography*, 2001.

[35] E. Damiani, S. D. C. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante. A reputation-based approach for choosing reliable resources in peer-to-peer networks. In *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS'02)*, 2002.

[36] M. Deutsch. Cooperation and trust: Some theoretical notes. In *Jones, M. R. (ed), Nebraska Symposium on Motivation.* Nebraska University Press, 1962.

[37] R. Dingledine, M. J. Freedman, D. Hopwood, and D. Molnar. A reputation system to increase mix-net reliability. In *Proceedings of the 4th International Workshop on Information Hiding*, 2001.

[38] R. Dingledine, N. Mathewson, and P. Syverson. Reputation in privacy enhancing technologies. In *Proceedings of the 12th Annual Conference on Computers, Freedom and Privacy*, 2002.

[39] R. Dingledine, N. Mathewson, and P. Syverson. Reputation in p2p anonymity systems. In *Proceedings of the Workshop on Economics of Peer-to-Peer Systems*, 2003.

[40] R. Dingledine, N. Mathewson, and P. F. Syverson. Tor: The second-generation onion router. In *Proceedings of the USENIX Security Symposium*, 2004.

[41] D. Donato, M. Paniccia, M. Selis, C. Castillo, G. Cortese, and S. Leonardi. New metrics for reputation management in p2p networks. In *Proceedings of the 3rd International Workshop on Adversarial Information Retrieval on the Web (AIRWeb'07)*, 2007.

[42] J. R. Douceur. The sybil attack. In *Proceedings of the First International Workshop on Peer-to-Peer Systems*, 2002.

[43] W. Du. *A Study of Several Specific Secure Two-Party Computation Problems.* PhD thesis, Purdue University, West Lafayette, IN, USA, 2001.

[44] Duedil. Duedil – transparent, constructive feedback on your profile. http://www.duedil.com/, July 2010.

[45] eBay. Upcoming changes to feedback. http://pages.ebay.com/services/forum/new.html, 2008. Retrieved June 30, 2008.

[46] T. ElGamal. A public-key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, IT-31(4):469472, 1985.

[47] R. Falcone and C. Castelfranchi. *Social Trust: A Cognitive Approach. In Trust and Deception in Virtual Societies.* Kluwer Academic Publishers, 2001.

[48] D. Ferraiolo and R. Kuhn. Role based access control. In *Proceedings of the 15th National Computer Security Conference*, pages 554 – 563, October 13 - 16 1992.

[49] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO'86*, 1986.

[50] A. Fog. Branch prediction in the pentium family. http://x86.org/articles/branch/branchprediction.htm, 2008. Retrieved February 24, 2008.

[51] N. Gal-Oz, E. Gudes, and D. Hendler. A robust and knot-aware trust-based reputation model. In *Proceedings of the Joint iTrust and PST Conferences on Privacy, Trust Management and Security (IFIPTM 2008)*, 2008.

[52] D. Gambetta. *Trust: Making and Breaking Cooperative Relatioins*, chapter Can We Trust Trust?, pages 213 – 237. Department of Sociology, University of Oxford, 2000.

[53] S. Garfinkel. *PGP: Pretty Good Privacy*. Number ISBN 1-56592-098-8. O'Reilly & Associates, 1991.

[54] J. Golbeck and J. Hendler. Filmtrust: Movie recommendations using trust in web-based social networks. In *Proceedings of the 3rd IEEE Consumer Communications and Networking Conference (CCNC 2006)*, Las Vegas, NV, USA, 2006.

[55] O. Goldreich. Secure multi-party computation. Working Draft, Version 1.4, 2002.

[56] O. Goldreich. *The Foundations of Cryptography - Volume 2*. Cambridge University Press, 2004.

[57] S. Goldwasser and S. Micali. Probabilistic encryption. *Special Issue of Journal of Computer and Systems Sciences*, 28(2):270–299, April 1984.

[58] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186208, 1989.

[59] T. Grandison and M. Sloman. A survey of trust in internet applications. *IEEE Communications Surveys and Tutorials*, 4, 2001.

[60] N. Griffiths. Task delegation using experience based multidimensional trust. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*, The Netherlands, 2005.

[61] E. Gudes, N. Gal-Oz, and A. Grubshtein. Methods for computing trust and reputation while preserving privacy. In *Proceedings of the 23rd Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, 2009.

[62] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proceedings of the International World Wide Web Conference (WWW 2004)*, 2004.

[63] M. Gupta, P. Judge, and M. Ammar. A reputation system for peer-to-peer networks. In *Proceedings of the 13th international Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2003.

[64] J. Haller. A bayesian reputation system for virtual organizations. In *Negotiation, Auctions, and Market Engineering*, 2008.

[65] M. Ham and G. Agha. Ara: A robust audit to prevent free-riding in p2p networks. In *Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing*, 2005.

[66] O. Hasan. Paillier cryptosystem – an interactive demo. http://liris.cnrs.fr/ ohasan/pprs/paillierdemo/index.html, June 2010.

[67] O. Hasan, J.-M. Pierson, and L. Brunie. Access control for ubiquitous environments based on subjectivity eliminated trust propagation. In *Proceedings of the 3rd International Symposium on TRUST, in conjunction with the 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing (EUC'08)*, page 603609, Shanghai, China, December 1720 2008.

[68] H. Hirsh, C. Basu, and B. D. Davison. Learning to personalize. *Communications of the ACM*, 43(8):102 – 106, August 2000.

[69] K. Hoffman, D. Zage, and C. Nita-Rotaru. A survey of attack and defense techniques for reputation systems. *ACM Computing Surveys*, 41(4), December 2009.

[70] J. Hornsby. *Actions*. Number ISBN 978-0710004529. Routledge, 1980.

[71] HP. Identity-based access control. Technical report, Hewlett-Packard, 2006. ProCurve Networking by HP.

[72] J. Hu and M. Burmester. Lars a locally aware reputation system for mobile ad hoc networks. In *Proceedings of the 44th Annual Southeast Regional Conference*, Melbourne, Florida, USA, 2006.

[73] L. X. Hung, P. D. Giang, Y. Zhung, T. V. Phuong, S. Lee, and Y.-K. Lee. A trust-based security architecture for ubiquitous computing systems. In *Proc. of the IEEE Intl. Conf. on Intelligence and Security Informatics (ISI 2006)*, San Diego, CA, USA, May 2006.

[74] I. Ioannidis, A. Grama, and M. Atallah. A secure protocol for computing dot-products in clustered and distributed environments. In *Proceedings of the 2002 International Conference on Parallel Processing*, 2002.

[75] C. M. Jonker and J. Treur. Formal analysis of models for the dynamics of trust based on experiences. In *Proceedings of the 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, 1999.

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

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

[78] L. Kagal, T. Finin, and A. Joshi. Trust-based security in pervasive computing environments. *IEEE Computer*, December 2001.

[79] E. A. Kaluscha. *The Importance of Initial Consumer Trust in B2C Electronic Commerce - A Structural Equation Modeling Approach.* PhD thesis, University of Klagenfurt, Austria, 2004.

[80] S. D. Kamvar, M. T. Schlosser, and H. GarciaMolina. The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the 12th International Conference on World Wide Web (WWW 2003)*, Budapest, Hungary, May 2003.

[81] M. Kinateder and S. Pearson. A privacy-enhanced peer-to-peer reputation system. In *Proceedings of the 4th International Conference on Electronic Commerce and Web Technologies*, 2003.

[82] M. Kinateder, R. Terdic, and K. Rothermel. Strong pseudonymous communication for peer-to-peer reputation systems. In *Proceedings of the 2005 ACM symposium on Applied computing*, 2005.

[83] L. Kissner. *Privacy-Preserving Distributed Information Sharing.* PhD thesis, Computer Science Department, Carnegie Mellon University, PA, USA, July 2006. CMU-CS-06-149.

[84] D. Kreps and R. Wilson. Reputation and imperfect information. *Journal of Economic Theory*, 27(2):253 279, 1982.

[85] P. Langley. Adaptive user interfaces and personalization. http://www.isle.org/ langley/adapt.html, 2008. Retrieved February 24, 2008.

[86] R. Levien. Attack resistant trust metrics. Manuscript, University of California - Berkeley. www.levien.com/thesis/compact.pdf, 2002.

[87] R. Levien. *Attack-Resistant Trust Metrics (Chapter 5). Computing with Social Trust.* Springer London, 2008.

[88] R. Levien and A. Aiken. Attack-resistant trust metrics for public key certification. In *Proceedings of the 7th USENIX Security Symposium*, San Antonio, Texas, January 26-29 1998.

[89] P. A. Loscocco, S. D. Smalley, P. A. Muckelbauer, R. C. Taylor, S. J. Turner, and J. F. Farrell. The inevitability of failure: The flawed assumption of security in modern computing environments. In *Proceedings of the 21st National Information Systems Security Conference*, page 303314, 1998.

[90] S. P. Marsh. *Formalising Trust as a Computational Concept*. PhD thesis, Department of Mathematics and Computer Science, University of Stirling, Scotland, UK, 1994.

[91] S. Marti and H. Garcia-Molina. Limited reputation sharing in p2p systems. In *Proceedings of the 5th ACM Conference on Electronic Commerce*, 2004.

[92] L. D. Martino, Q. Ni, D. Lin, and E. Bertino. Multi-domain and privacy-aware role based access control in ehealth. In *Proceedings of the Second International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth 2008)*, pages 131–134, 2008.

[93] M. McCord and P. Ratnasingam. The impact of trust on the technology acceptance model in business to consumer e-commerce. In *Proceedings of the International Conference of the Information Resources Management Association: Innovations Through Information Technology*, New Orleans, LA, USA, May 2004.

[94] D. H. McKnight and N. L. Chervany. The meanings of trust. Technical Report MISRC Working Paper Series 96-04, University of Minnesota, Management Information Systems Research Center, 1996.

[95] D. H. McKnight, V. Choudhury, and C. Kacmar. Developing and validating trust measures for e-commerce: An integrative typology. *Information Systems Research*, 13(3):334 – 359, September 2002.

[96] D. H. McKnight, L. L. Cummings, and N. L. Chervany. Initial trust formation in new organizational relationships. *Acad. Management Rev.*, 23(3):473490, 1998.

[97] L. McNamara, C. Mascolo, and L. Capra. Media sharing based on colocation prediction in urban transport. In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, 2008.

[98] K. Meng, X. Zhang, X. chun Xiao, and G. du Zhang. A bi-rating based personalized trust management model for virtual communities. In *Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control (ICNSC '06)*, 2006.

[99] C. Mitchell, editor. *Trusted computing*. Institution of Electrical Engineers, 2005.

[100] National Institute of Standards and Technology (NIST). Nist/sematech e-handbook of statistical methods - percentiles. http://www.itl.nist.gov/div898/handbook/prc/section2/prc252.htm, 2008. Retrieved February 24, 2008.

[101] J. Nin, B. Carminati, E. Ferrari, and V. Torra. Computing reputation for collaborative private networks. In *Proceedings of the 33rd Annual IEEE International Computer Software and Applications Conference*, 2009.

[102] T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. In *Advances in Cryptology – EUROCRYPT'98*, 1998.

[103] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford University, January 29 1998.

[104] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques*, 1999.

[105] E. Pavlov, J. S. Rosenschein, and Z. Topol. Supporting privacy in decentralized additive reputation systems. In *Proceedings of the Second International Conference on Trust Management (iTrust 2004)*, Oxford, UK, 2004.

[106] S. Pearson and B. Balacheff, editors. *Trusted Computing Platforms: TCPA Technology in Context*. Prentice Hall, 2003.

[107] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology*, 1991.

[108] F. Pingel and S. Steinbrecher. Multilateral secure cross-community reputation systems for internet communities. In *Proceedings of the Fifth International Conference on Trust and Privacy in Digital Business (TrustBus 2008)*, 2008.

[109] J. A. Pouwelse, P. Garbacki, D. H. J. Epema, and H. J. Sips. The bittorrent p2p file-sharing system: Measurements and analysis. In *Peer-to-Peer Systems IV*, 2005.

[110] D. Quercia. *Trust Models for Mobile Content-Sharing Applications*. PhD thesis, University College London, 2009.

[111] 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. Michael R. Baye, editor. Volume 11 of Advances in Applied Microeconomics*, pages 127–157, 2002.

[112] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara. Reputation systems. *Communications of the ACM*, 43(12):4548, December 2000.

[113] M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *Proceedings of the Second International Semantic Web Conference*, pages 351–368, 2003.

[114] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120126, 1978.

[115] R. L. Rivest and A. T. Sherman. Randomized encryption techniques. In *CRYPTO'82*, 1982.

[116] J. B. Rotter. A new scale for the measurement of interpersonal trust. *Journal of Personality*, 35(4):651 – 665, December 1967.

[117] J. B. Rotter. Generalized expectancies for interpersonal trust. *American Psychologist*, 26(5):443 – 452, 1971.

[118] R. Saadi, O. Hasan, J.-M. Pierson, and L. Brunie. Establishing trust beliefs based on a uniform disposition to trust. In *Proceedings of the 3rd IEEE International Conference on Signal-Image Technology & Internet-Based Systems (SITIS 2007)*, page 221228, Shanghai, China, December 16-19 2007.

[119] R. Saadi, J.-M. Pierson, and L. Brunie. Authentication and access control using trust collaboration in pervasive grid environment. In *Proceedings of the International Conference on Grid and Pervasive Computing (GPC 2007)*, 2007.

[120] P. Sant and C. Maple. A graph theoretic framework for trust – from local to global. *Information Visualization*, July 2006.

[121] L. J. Savage. *The Foundations of Statistics*. Dover Publications, 1972.

[122] S. Schiffner, S. Clau, and S. Steinbrecher. Privacy and liveliness for reputation systems. In *Proceedings of the Sixth European Workshop on Public Key Services, Applications and Infrastructures (EuroPKI'09)*, 2009.

[123] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612613, 1979.

[124] K. Shin and H. Yasuda. Practical anonymous access control protocols for ubiquitous computing. *Journal of Computers*, 1(8), December 2006.

[125] M. Srivatsa, L. Xiong, and L. Liu. Trustguard: Countering vulnerabilities in reputation management for decentralized overlay networks. In *Proceedings of the 14th International Conference on World Wide Web*, Chiba, Japan, 2005.

[126] S. Steinbrecher. Design options for privacy-respecting reputation systems within centralised internet communities. In *Security and Privacy in Dynamic Environments*, 2006.

[127] J. Steinhauer. Verdict in myspace suicide case. The New York Times, http://www.nytimes.com/2008/11/27/us/27myspace.html, November 2008.

[128] 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 2001Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 149–160, 2001.

[129] A. Swaminathan, R. G. Cattelan, C. V. Mathew, Y. Wexler, and D. Kirovski. Relating reputation and money in on-line markets. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, 2009.

[130] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557 – 570, October 2002.

[131] S. Toivonen, G. Lenzini, and I. Uusitalo. Context-aware trust evaluation functions for dynamic reconfigurable systems. In *Proceedings of the WWW'06 Workshop on Models of Trust for the Web (MTW'06)*, May 2006.

[132] H. Tran, M. Hitchens, V. Varadharajan, and P. Watters. A trust based access control framework for p2p file-sharing systems. In *Proceedings of the 38th Hawaii International Conference on System Sciences*, 2005.

[133] Unvarnished. Unvarnished – community-contributed reviews for business professionals. http://www.getunvarnished.com/, July 2010.

[134] J. Vaidya and C. Clifton. Privacy-preserving data mining: Why, how, and when. *IEEE Security and Privacy*, 2(6):19–27, November 2004.

[135] P. Victor, M. D. Cock, and C. Cornelis. *Trust and Recommendations (Chapter 20). Recommender Systems Handbook.* Springer, 2010.

[136] M. Voss, A. Heinemann, and M. Muhlhauser. A privacy preserving reputation system for mobile information dissemination networks. In *Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM)*, 2005.

[137] Y. Wang and J. Vassileva. Bayesian network trust model in peer-to-peer networks. In *Agents and Peer-to-Peer Computing*, 2004.

[138] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393, June 1998.

[139] M. Winslett, T. Yu, K. E. Seamons, A. Hess, J. Jacobson, R. Jarvis, B. Smith, and L. Yu. Negotiating trust on the web. *IEEE Internet Computing*, 6(6):30–37, November/December 2002.

[140] A. C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, 1982.

[141] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman. Sybilguard: Defending against sybil attacks via social networks. In *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM'06)*, pages 267–278. ACM Press, September 2006.

[142] L. Yu. A reputation system for bittorrent peer-to-peer file-sharing networks. Master's thesis, University of Wollongong, Australia, 2006.

[143] G.-X. Yue, F. Yu, Y.-J. Chen, R.-H. Wu, J.-Y. An, and R.-F. Li. Access control model for p2p file-sharing systems based on trust and recommendation. In *Proceedings of the Second International Conference on Semantics, Knowledge, and Grid (SKG'06)*, 2006.

[144] R. Zhou and K. Hwang. Powertrust: A robust and scalable reputation system for trusted peer-to-peer computing. *IEEE Transactions on Parallel and Distributed Systems*, 18(4):460–473, April 2007.

[145] C.-N. Ziegler and G. Lausen. Spreading activation models for trust propagation. In *Proceedings of the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE '04)*, 2004.