Preserving Privacy of Feedback Providers in Decentralized Reputation Systems

Omar Hasan[a,*], Lionel Brunie[a], Elisa Bertino[b]

[a] *University of Lyon, CNRS, INSA-Lyon, LIRIS, UMR5205, F-69621, France*
[b] *Department of Computer Science, Purdue University, IN 47907, USA*

Abstract

Reputation systems make the users of a distributed application accountable for their behavior. The reputation of a user is computed as an aggregate of the feedback provided by other users in the system. Truthful feedback is clearly a prerequisite for computing a reputation score that accurately represents the behavior of a user. However, it has been observed that users often hesitate in providing truthful feedback, mainly due to the fear of retaliation. We present a decentralized privacy preserving reputation protocol that enables users to provide feedback in a private and thus uninhibited manner. The protocol has linear message complexity, which is an improvement over comparable decentralized reputation protocols. Moreover, the protocol allows users to quantify and maximize the probability that their privacy will be preserved.

Keywords: reputation, privacy, trust, secret sharing, decentralization

1. Introduction

In recent years, reputation systems have gained popularity as a solution for securing distributed applications from misuse by dishonest users. A reputation system computes the reputation score of a user as an aggregate of the feedback provided by fellow users. Good behavior is rewarded by positive feedback and consequently a high reputation score. On the contrary, bad behavior results in negative feedback and a low reputation score, which can lead to isolation or exclusion from the application. Some examples of applications of reputation systems are as follows:

- According to a survey on fraud in e-commerce [1], fraud accounted for a total loss of US$ 2.7 billion in the United States and Canada in 2010. Reputation systems used by e-commerce websites (such as `ebay.com`,

*Corresponding author
Email addresses: `omar.hasan@insa-lyon.fr` (Omar Hasan), `lionel.brunie@insa-lyon.fr` (Lionel Brunie), `bertino@cs.purdue.edu` (Elisa Bertino)

amazon.com) mitigate the risk that a seller would turn out to be fraudulent.

- Several cases have been reported where fake online persona have hijacked the identity of professionals and then succeeded in connecting to their real network of acquaintances [2]. Reputation systems that root out fake profiles on social networks include Unvarnished [3] and Duedil [4].

- There is a risk in peer-to-peer file sharing networks (such as BitTorrent) that a file uploaded by a seeder is fake. Reputation systems for defeating fake content in peer-to-peer file sharing networks have been proposed by Costa and Almeida [5], Yu [6], and Kamvar et al. (EigenTrust) [7].

- Nodes in Mobile Ad-hoc Networks (MANETs) depend on neighbors to route their messages. However, neighbors may be selfish and may drop messages to conserve their resources. Reputation systems for discouraging selfish behavior in mobile ad-hoc networks include those by Hu and Burmester [8], and Buchegger et al. [9, 10].

Reputation score is an aggregate of the feedback, therefore an accurate reputation score is possible only if the feedback is accurate. However, it has been observed that the users of a reputation system often avoid providing truthful feedback [11]. This is particularly true about negative feedback. 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 lack of truthful feedback is computing reputation scores in a privacy preserving manner. A privacy preserving protocol for computing reputation scores operates such that the individual feedback of any user is not revealed. The implication is that the feedback provider is rendered uninhibited to provide truthful feedback.

Decentralized environments include decentralized social networks, peer-to-peer networks, and mobile ad-hoc networks. These environments are characterized by the absence of a trusted central authority. In Section 5, we observe that existing decentralized privacy preserving reputation protocols are either inefficient or depend on constructs such as Trusted Third Parties (TTPs), trusted hardware modules, or anonymous routing, which are often not feasible.

In this article, we propose a novel privacy preserving reputation protocol called the k-shares protocol, which is decentralized as well as efficient. The protocol requires only $O(n)$ number of messages for the computation of a reputation score, where n is the number of feedback providers. The protocol is secure under the standard semi-honest adversarial model (described in Section 2.2) and does not require TTPs, trusted hardware modules, or anonymous routing. Another novel aspect of our protocol is that a feedback provider is able to quantify and minimize the risk to its privacy before submitting feedback.

2. Framework

2.1. Agents, Trust, and Reputation

We model our environment as a multi-agent environment. An agent represents a user. Let $\mathbb{A}$ denote the set of all agents in the environment. $|\mathbb{A}| = N$.

We subscribe to the definition of trust by sociologist Diego Gambetta [12], which characterizes trust as binary-relational, directional, contextual, and quantifiable as subjective probability. Our formal definition of trust attempts to capture each of these characteristics.

Let $\mathbb{D}$ denote an asymmetric binary relation on the set $\mathbb{A}$. Let $\mathbb{T} \subseteq \mathbb{D}$ be the set of all existing trust relationships between agents. $(a, b) \in \mathbb{T}$, where $a, b \in \mathbb{A}$, implies that an agent $\boldsymbol{a}$ has a trust relationship towards an agent $\boldsymbol{b}$. The asymmetric binary relation $\mathbb{D}$ captures the directional and binary-relational characteristics of trust.

Let Ψ denote the set of all actions. Examples of actions include: "prescribe correct medicine", "repair car", "deliver product sold online", "upload authentic content", "route a message for a neighbor", etc.

Let *perform* denote a function, such that $perform : \mathbb{T} \times \Psi \rightarrow \{true, false\}$. The function $perform(a, b, \psi)$ 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. 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.

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

In Definition 1, action ψ is the context of trust, and the subjective probability $P(perform(a, b, \psi) = true)$ is the quantification of trust. When the context of trust is clear, we adopt the simplified notation l_{ab} for $P(perform(a, b, \psi) = true)$. Table 1 summarizes the various elements of the definition of trust.

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. The set of all source agents of an agent $\boldsymbol{b}$ in context ψ is given as $S_{b,\psi}$. The simplified notation S_b is used instead of $S_{b,\psi}$ when the context is clear. We also refer to the quantification of a source agent $\boldsymbol{a}$'s trust in agent $\boldsymbol{b}$ as feedback.

Definition 2. ***Reputation.*** *Let $S_t = \{a_1 \dots a_n\}$ be the set of source agents of an agent t in context ψ. Let rep denote a function, such that $rep : [0, 1]_1 \times \dots \times [0, 1]_n \rightarrow \mathbb{R}$. Then the reputation of agent t in context ψ is given as: $rep(P(perform(a_1, t, \psi) = true), \dots, P(perform(a_n, t, \psi) = true))$, or in simplified notation: $rep(l_{a_1 t}, \dots, l_{a_n t})$.*

We have defined the reputation of an agent as any function that aggregates the feedback of its source agents. The reputation of an agent t is denoted by $r_{t,\psi}$, or r_t when the context is clear.

Let a function $rep_\oplus$ be an implementation of the reputation of an agent t in context ψ.

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

Table 1: Elements of the definition of trust.

Element	Description
$a \ \mathbb{T} \ b$	Agent $\boldsymbol{a}$ has a directional trust relationship towards agent $\boldsymbol{b}$. Relation $\mathbb{T} \subseteq \mathbb{D}$, where $\mathbb{D}$ is an asymmetric binary relation on set $\mathbb{A}$, the set of all agents in the environment.
ψ	An action, which is the context of agent $\boldsymbol{a}$'s trust in agent $\boldsymbol{b}$.
$perform(a, b, \psi)$	$perform : \mathbb{T} \times \Psi \rightarrow \{true, false\}$. $\psi \in \Psi$. The function $perform(a, b, \psi)$ outputs $true$ if agent $\boldsymbol{b}$ performs the action ψ anticipated by agent $\boldsymbol{a}$, otherwise it outputs false.
$P(perform(a, b, \psi) = true)$	The subjective probability (according to agent $\boldsymbol{a}$) that agent $\boldsymbol{b}$ will perform the action ψ. It is the quantification of agent $\boldsymbol{a}$'s trust in agent $\boldsymbol{b}$. It is abbreviated as l_{ab} when the context of trust is clear.

The function $rep_{\oplus}$ implements the reputation of an agent t as the mean of the feedback values of its source agents. 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. We derive the mean from the sum in order to normalize the reputation values.

Definition 3. ***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: a querying agent* q*, a target agent* t*, and all* n *source agents of* t *in the context* ψ*, 2) the inputs include: the feedback 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*.*

2.2. Adversary

We refer to the coalition of dishonest agents as the adversary. In this paper, we propose a solution for the semi-honest adversarial model. The agents in this model always execute the protocol according to the specification. 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 gleaned during the execution of the protocol.

2.3. Privacy

Definition 4. ***Private Data.*** *Let* x *be some data and an agent* $\boldsymbol{a}$ *be the owner of* x*. Then* x *is agent* $\boldsymbol{a}$*'s private data if agent* $\boldsymbol{a}$ *desires that no other agent learns* x*. An exception is those agents to whom* $\boldsymbol{a}$ *reveals* x *herself. However, if* $\boldsymbol{a}$ *reveals* x *to an agent* $\boldsymbol{b}$*, then* $\boldsymbol{a}$ *desires that* $\boldsymbol{b}$ *does not use* x *to infer more information. Moreover,* $\boldsymbol{a}$ *desires that* $\boldsymbol{b}$ *does not reveal* x *to any third party.*

Definition 5. ***Preservation of Privacy (by an Agent).*** *Let x be an agent $\boldsymbol{a}$'s private data that agent $\boldsymbol{a}$ reveals to an agent $\boldsymbol{b}$. Then agent $\boldsymbol{b}$ is said to preserve the privacy of agent $\boldsymbol{a}$ w.r.t. x, if 1) $\boldsymbol{b}$ does not use x to infer more information, and 2) $\boldsymbol{b}$ does not reveal x to any third party.*

Let action ρ = "preserve privacy". The action ρ 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.

Definition 6. ***Trusted Third Party (TTP).*** *Let $S \subseteq \mathbb{A}$ be a set of n agents, and $TTP_S \in \mathbb{A}$ be an agent. Then TTP_S is a Trusted Third Party (TTP) for the set of agents S if for each $a \in S$, $P(perform(a, TTP_S, \rho) = true) = 1$.*

We define *security threshold* as 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. We consider as *high* any probability greater than or equal to the *security threshold*, and as *low* any probability less than $1-$ *security threshold*.

We adopt the Ideal-Real approach [13] to define privacy preserving reputation protocols.

Definition 7. ***Ideal Privacy Preserving Reputation Protocol.*** *Let Π be a reputation protocol (Definition 3). 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, 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 $P(perform(a, TTP_{S_t}, \rho) = true) = 1$.

Definition 8. ***Real Privacy Preserving Reputation Protocol.*** *Let $\mathcal{I}$ be an ideal privacy preserving reputation protocol (Definition 7). Then $\mathcal{R}$ is a real privacy preserving reputation protocol w.r.t. $\mathcal{I}$, if: 1) $\mathcal{R}$ has the same parameters (participants, private inputs, output, adversary, etc.) as $\mathcal{I}$, except that there is no TTP_{S_t} as a participant 2) with high probability, the adversary learns no more information about the private input of any agent $\boldsymbol{a}$ than it can learn in protocol $\mathcal{I}$.*

2.4. Problem Definition

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 realized as $rep_{\oplus}$. Agents q, t, and m of the source agents are considered to be dishonest, where $m < n$. However, q wishes to learn the correct output and therefore does not take any actions that alter the output. The reputation protocol Π is required to be decentralized and secure under the semi-honest model.

3. The *k*-Shares Reputation Protocol

The k-shares protocol is inspired in part by Pavlov et al. [14, Section 5.2]. However, our protocol has a lower message complexity of $O(n)$ as opposed to $O(n^2)$ of the protocol in [14]. Moreover, our protocol allows agents to quantify and maximize the probability that their privacy will be preserved before they submit their feedback. The important steps of the protocol are outlined below.

1. **Initiate.** 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. 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 $U_a = \{u_{a,1} \dots u_{a,k_a}\}$, where $1 \leq k_a \leq k$. Agent $\boldsymbol{a}$ selects these agents such that: $P(perform(a, u_{a,1}, \rho) = false) \times \dots \times P(perform(a, u_{a,k_a}, \rho) = false)$ is low. That is, the probability that all of the selected agents will collude to break agent $\boldsymbol{a}$'s privacy is low. k is a constant, such that $k \ll n$.
3. **Prepare Shares.** Agent $\boldsymbol{a}$ then prepares $k_a + 1$ shares of its secret feedback value l_{at}. The shares, given as: $x_{a,1} \dots x_{a,k_a+1}$, 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, such that the sum of the shares is equal to the feedback value.
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 each share $x_{a,i}$ 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 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 $r_{t,\psi} = (\sum_{a \in S_t} \sigma_a)/n$.

3.1. Protocol Specification

The protocol is specified in Figure 1. The function $set\_of\_trustworthy(a, S)$ 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: $P(perform(a, u_{a,1}, \rho) = false) \times \dots \times P(perform(a, u_{a,k_a}, \rho) = false)$ is low, with the minimum possible k_a.

3.2. Security Analysis

3.2.1. Correctness

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: $\sum_{i=1}^{n} l_{a_i t}$. Thus, 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.

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: $r = \sum_{i=1}^{n} \sigma_{a_i} = \sum_{i=1}^{n}(\sum_{j=1}^{k_{a_i}+1} x_{a_i,j}) = \sum_{i=1}^{n} l_{a_i t}$. From Definition 2 and Equation 1, we can infer that $r_{t,\psi} = r/n$ computed by q is the correct reputation of agent t in context ψ.

3.2.2. Privacy of Feedback Values

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 $[-X, X]$. 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 l_{at}. 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 adversary can learn the $k_a + 1$ shares of an agent $\boldsymbol{a}$ from the protocol.

Agent $\boldsymbol{a}$ sends each share $x_{a,i}$ only to agent $u_{a,i}$, where $i \in \{1 \dots k_a\}$. 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: $P(perform(a, u_{a,i}, \rho) = false)$.

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

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: $P(perform(a, u_{a,1}, \rho) = false) \times \ldots \times P(perform(a, u_{a,k_a}, \rho) = false)$. If we consider that each agent $a \in S_t$ selects the agents $u_{a,1} \ldots u_{a,k_a}$ such that this probability is low, then 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. From Definition 7 and Definition 8, we can infer that the k-shares reputation protocol is a real privacy preserving reputation protocol.

3.2.3. Notes on the Cardinality of the Sets S_t, J_a, and U_a

The set S_t is the set of source agents who provide feedback about a target agent t. The protocol requires that $k \ll n$, where $n = |S_t|$. That is, the constant k is significantly less than the cardinality of the set S_t. As we observe in Section 3.3, the condition $k \ll n$ serves to limit the complexity of the protocol to $O(n)$ in terms of the number of messages exchanged. However, this implies that the protocol is suitable only for determining reputation of target agents with a large number of feedback providers in relation to k. This may not be a disadvantage since the low complexity of the protocol would be most advantageous for higher values of n in comparison to protocols such as the one by Pavlov et al. which has a complexity greater than $O(n^2)$.

The adversary can ascertain the interval of the submitted feedback values from publicly available information, which includes the output of the protocol (the reputation r_t computed as the mean of the feedback values) and the cardinality n of the set S_t. The interval of the submitted feedback values, $[\min_{a \in S_t} l_{at}, \max_{a \in S_t} l_{at}]$, can be computed as follows:

$$\left[\begin{array}{rl} 1 - ((1 - r_t) \times n) & \text{if } 1 - ((1 - r_t) \times n) \geq 0 \\ 0 & \text{if } 1 - ((1 - r_t) \times n) < 0 \end{array} , \begin{array}{rl} r_t \times n & \text{if } r_t \times n \leq 1 \\ 1 & \text{if } r_t \times n > 1 \end{array} \right] \quad (2)$$

A combination of higher values of n and values of r_t closer to the center of $[0, 1]$ increases the size of the interval, thus reducing the amount of information gleaned by the adversary. When the interval determined by the adversary is $[0, 1]$, the adversary gains no additional information about the submitted feedback values since $[0, 1]$ is the universal interval for all feedback values. Some examples of the effects of the values of r_t and n are given in Table 2.

We note that the above information can be ascertained by the adversary in the real privacy preserving reputation protocol as well as in the ideal protocol since the adversary uses only the publicly available information. The cardinality of the set S_t therefore does not have an effect specifically on the k-shares protocol in terms of preserving the privacy of the feedback values. In fact, since the k-shares protocol is not meant to be used for low values of n, the adversary is unlikely to learn a small interval, unless $r_t \approx 0$ or $r_t \approx 1$.

Table 2: Interval of the submitted feedback values.

r_t	n	**Interval**
0	5	$[0, 0]$
0.2	2	$[0, 0.4]$
0.2	5	$[0, 1]$
0.6	2	$[0.2, 1]$
0.6	5	$[0, 1]$
0.8	2	$[0.6, 1]$
0.8	5	$[0, 1]$
1	5	$[1, 1]$

The set J_a is the set of fellow source agents in the protocol from whom an agent $\boldsymbol{a}$ receives shares. The cardinality of the set J_a determines whether agent q learns the final share x_{a,k_a+1} of agent $\boldsymbol{a}$ or not. Agent $\boldsymbol{a}$ is required to send $\sigma_a = x_{a,k_a+1} + \sum_{v \in J_a} x_v$ to q. If the set J_a is not empty then σ_a is the sum of x_{a,k_a+1} and the shares received. This implies that q cannot distinguish the value of the share x_{a,k_a+1}. However, if the set J_a is empty then $\sigma_a = x_{a,k_a+1}$ and q learns its value.

As we note in the previous section, the revelation of the final share x_{a,k_a+1} of agent $\boldsymbol{a}$ has no bearing on the privacy of $\boldsymbol{a}$'s feedback value. The share x_{a,k_a+1} serves to hide the value of the share received, in case only one share is received, that is, $|J_a| = 1$.

The set U_a is the set of agents selected by an agent a to send shares of private feedback. We discuss the implications of the cardinality of the set U_a in the following section.

3.2.4. Privacy of Trust Relationships

In the protocol specified in Figure 1, 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 are selected randomly from the remaining source agents in S_t. The consequence of this extension is that the set U_a no longer consists exclusively of agents whom agent $\boldsymbol{a}$ considers trustworthy. Thus agent q cannot ascertain whether an agent in U_a is agent $\boldsymbol{a}$'s trustworthy agent or an agent that has been randomly selected. Adding the random potentially untrustworthy agents to the set U_a does not weaken the security of agent $\boldsymbol{a}$ since all agents in the set U_a must collude to learn its private feedback value.

An adversary may attempt repeated queries to determine the agents that occur most frequently in U_a, thus revealing $\boldsymbol{a}$'s trustworthy agents. However, this attack can be countered if agent $\boldsymbol{a}$ selects an identical set of agents for each repeated query.

It is possible that an agent $\boldsymbol{a}$'s security threshold is satisfied with as low as one trustworthy agent in U_a. This implies that the adversary can only make a random guess with a probability $1/k$ of being correct that a given agent in the set U_a is one of $\boldsymbol{a}$'s trustworthy agents.

3.2.5. Abstaining when Risk to Privacy is High

The privacy of the k-shares protocol depends on the assumption that each agent $a \in S_t$ will find trustworthy agents in S_t. However, the protocol may be extended such that agents are allowed to abstain when they do not 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 along with any shares received added to it. The abstaining source agent would also notify the querying agent that it has abstained. The querying agent would compute reputation as the sum of all shares divided by the number of non-abstaining agents. In section 4.2, we observe that the protocol computes sufficiently accurate reputation scores even if a large number of agents abstain.

It can be inferred about an abstaining agent that it does not sufficiently trust any of the fellow source agents in the protocol. To prevent such inference, an agent can adopt a policy of permanently abstaining for a certain percentage of target agents even if their set of sources contains trustworthy agents. This prevents the adversary from deterministically learning whether the agent is abstaining because it did not find trustworthy agents or because it is taking privacy preserving measures.

3.2.6. An Attack on the Ideal Protocol

We describe an attack in which the adversary attempts to determine the private feedback of a source agent over the course of two reputation queries. Consider the scenario when a new agent $\boldsymbol{a}$ is added to the set of source agents S_t. Let $S_t^{'} = S_t \cup \{a\}$. Let the reputation of the target agent t be r_t and $r_t^{'}$ for the set of source agents S_t and $S_t^{'}$ respectively. A querying agent q that queries the reputation of the target agent t with both sets of source agents can compute the private feedback of agent $\boldsymbol{a}$ as $l_{at} = (r_t^{'} \times (n+1)) - (r_t \times n)$, where $n = |S_t|$. A similar attack can also determine the private feedback of an existing source agent that drops out from the set of source agents.

The ideal protocol (Definition 7) is vulnerable to this attack. Consequently, the k-shares protocol is also vulnerable as it emulates the ideal protocol. We note that this is a general issue for all protocols that can produce the sum of the feedback values of the following sets of source agents: S_t, and $S_t \cup \{a\}$ or $S_t / \{b\}$, where $a \notin S_t$ and $b \in S_t$. We discuss three different strategies for countering this attack and list their individual tradeoffs.

Random Number Generator. We have previously presented an additive privacy preserving reputation protocol [15] based on secure sum [16], for which we addressed this attack using the data perturbation technique. The interval for feedback is $[-1, 1]$. A trusted random number generator generates a random number $x \in [-Y, Y]$. It then generates random

numbers $x_1, x_2, \ldots x_{n-1}$, and computes x_n such that $x = \sum_{i=1}^{n} x_i$, where $Y > 1$. It delivers each number x_i to source agent a_i. Each source agent a_i adds the random number to its feedback before submitting it to the protocol. The sum of the feedback values is perturbed by a new random number in $[-Y, Y]$ for each query. Thus, the adversary cannot use this attack to deterministically measure the difference between any two sums, introduced due to the addition or subtraction of the feedback of a certain source agent. The tradeoff is the need for a random number generator that is trusted by the source agents in the context of preserving privacy.

Trusted Subset. A new source agent abstains from submitting feedback until it becomes a member of a *trusted subset*. A *trusted subset* is formed by two or more new source agents who trust each other in the context of preserving privacy. The agents in a trusted subset submit feedback in tandem, that is, they either all submit their feedback or none of them does. If one of the members of the subset leaves the set of source agents, then all the other members of the subset leave as well. A leaving member may re-enter and form a new trusted subset. Membership in a trusted subset is mutually exclusive to other trusted subsets, that is, one agent belongs to only one trusted subset.

The effect achieved is that the attack can reveal only the sum of the feedback of the agents in a trusted subset. The adversary can no longer learn individual feedback of an agent through this attack as long as the fellow members in its trusted subset are honest. The absence or presence of the feedback of one honest agent would imply the same for the feedback of the other agents in its trusted subset. Thus, if one honest agent is added or dropped from the set of source agents, the difference in the before and after reputation scores would equal the sum of the feedback of all the agents in its trusted subset. A drawback of this approach is that the members of a trusted subset have to reveal that they trust their fellow members in terms of preserving privacy.

The given attack is addressed by Kerschbaum [17] as follows: A central reputation manager (termed the service provider) updates the score of a target agent only after a certain number of new individual feedback values have been submitted. The solution by Kerschbaum achieves an effect which is similar to that in our solution. However, as recognized by Kerschbaum, their solution is vulnerable to intentional new feedback submitted by dishonest colluding agents. On the contrary, our solution does not suffer from this issue since an agent relies only on other agents that it trusts.

Probabilistic Participation. A system wide probability threshold p is defined. Each honest source agent submits his feedback if the value of a local random variable is within p, otherwise he abstains. The effect achieved is that for each query only a random $p\%$ (on average) of the source agents will submit their feedback. Let's consider the probability that after the com-

putation of reputation with a set of non-abstaining honest source agents S_t, the set $S_t \cup \{a\}$ or $S_t/\{b\}$ (where $a \notin S_t$ and $b \in S_t$) will occur as the set of non-abstaining honest source agents in a subsequent query. This probability can be quiet low depending on the value of p, the cardinality of S_t, and the number of subsequent queries permitted. We propose to analyze this probability in our future work on a protocol for the malicious adversarial model (discussed in Section 6.1). We can imagine a protocol that lowers this probability, for example, by choosing a favorable value of p and by limiting the queries permitted per querying / target agent pair. Additionally, the protocol could place a cost on each query such that running multiple queries (even by multiple colluding agents) to mount this attack becomes an expensive proposition. The tradeoff of probabilistic participation is the accuracy of the reputation score. However, we have previously observed in [18, Section 4.8.3](using the Advogato.org dataset) that if reputation is computed as mean, then a significant percentage of source agents can abstain without significantly compromising accuracy.

To the best of our knowledge, other decentralized additive reputation systems in the literature (for example, Pavlov et al. [14] and Gudes et al. [19]) have not addressed the described attack.

3.3. Complexity Analysis

A detailed analysis of the number of messages and the amount of information exchanged in the k-shares protocol is given in Table 3. Considering the assumption that $k \ll n$, 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: $O(n)$).

Table 3: Complexity analysis.

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 $k \ll n$

3.4. Discussion

The protocol by Pavlov et al. [14] requires each agent to send shares to all other $n-1$ source agents in the protocol. The result is $O(n^2)$ messages. We argue that sending shares to $n-1$ potentially unknown agents is not productive

since they can all turn out to be dishonest. Instead, in the k-shares protocol, each agent $\boldsymbol{a}$ relies on at most k agents who are selected based on $\boldsymbol{a}$'s knowledge of their trustworthiness. The advantages are twofold. Firstly, an agent is able to quantify and maximize the probability that its privacy will be preserved. This also allows us to extend the protocol such that an agent can abstain from providing feedback if the risk to its privacy is high. Secondly, limiting the number of shares to $k \ll n$, results in a protocol that requires only $O(n)$ messages. We observe in the next section that the privacy of a high majority of agents can be assured with k as small as 2. Increasing k to $n-1$ gives no significant advantage.

4. Experiments

4.1. The Dataset: Advogato.org

We use the real web of trust of Advogato.org 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%.

The members of Advogato are expected to not post spam, not attack the Advogato trust metric, etc. We therefore argue that the context "be a responsible member of the open source software developer community" comprises of the context "be honest". 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 experience with Advogato.

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 security threshold to be satisfied.

4.2. Experiment 1

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 $\{5, 10, 15, 20, 25, 50, 75, 100, 500\}$.

Results: For $min = 25$, we observe that the assumption holds for 81.7% of instances of source agents. Additionally, 85.8% for $min = 50$, 87.0% for

$min = 75$, 87.4% for $min = 100$, and 87.5% for $min = 500$. We note that the increase in the percentage is significant up to $min = 100$. This is due to the greater choice of trustworthy agents available for each agent when the protocol has more source agents. At $min = 5$, the percentage is 72.5%, which implies that approximately 30% of the source agents will have to abstain. However, in a separate experiment (full details not included due to space limitation), we observed that at $min = 25$, even if only around 40% of agents participate, over 95% of the computed reputation scores have an error of at most 0.1 compared to the true scores. Additionally, over 85% at $min = 10$, and over 90% at $min = 15$. Thus, even a significant portion of agents abstaining does not pose an issue.

4.3. Experiment 2

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 run 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%. The rise 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. Increasing k over 2, even up to 500, does not result in a significant advantage (86.3% at $k = 500$). Thus, in this dataset, privacy can be preserved for a high percentage of source agents with k as small as 2.

5. Related Work

Secure sum [16] is a well-known secure multi-party computation protocol [13], which computes the sum of inputs from distributed sites while preserving their privacy. The secure sum protocol is suitable as a privacy preserving reputation protocol because inputs can be considered as feedback values about a certain target entity and the output sum can be considered as the entity's reputation. The weakness of the secure sum protocol is that it cannot preserve privacy if the sites collude with each other. In contrast, our k-shares protocol is secure under the standard semi-honest adversarial model, that is, it preserves privacy even if sites collude. Both secure sum and k-shares require $O(n)$ number of messages, where n is the number of entities with inputs.

Pavlov et al. [14] propose a decentralized privacy preserving reputation protocol for the semi-honest adversarial model. The protocol comprises of two steps: 1) The first step is the execution of a witness (feedback provider) selection scheme, which guarantees the inclusion of a certain number of honest witnesses as participants. 2) The second step is the decentralized computation of the reputation as the sum of the feedback values. According to our analysis, the protocol for the semi-honest model requires an exchange of $O(N) + O(n^2)$ number of messages, where N is the number of potential witnesses and n is the

number of witnesses selected. The witness selection scheme requires $O(N)$ messages. The decentralized computation of the reputation uses a secret sharing scheme in which every witness sends a message to all other $n-1$ participating witnesses thus resulting in a total exchange of $O(n^2)$ messages. Our k-shares protocol is inspired in part by the protocol by Pavlov et al. However, our protocol requires only $O(n)$ messages as opposed to $O(N)+O(n^2)$ required by Pavlov et al. The reason for the lower complexity is that each entity in our protocol selects up to k (where $k \ll n$) fellow entities based on subjective trust (thus eliminating the need for a costly witness selection scheme) and sends messages to only those k entities (thus requiring only a linear number of messages instead of quadratic). Another key difference is that our protocol allows entities to quantify and minimize the risk to their privacy before feedback is submitted.

Gudes et al. [19] present three protocols that augment their Knots reputation system [20] with privacy preserving features for security under the semi-honest adversarial model. The Knots reputation system is a personalized reputation system, which implies that feedback is collected only from the entities whom the querying entity trusts. The complexity of the first two privacy preserving protocols is $O(t)$, where t is the number of querying entity's trusted entities who have feedback about the target. Our k-shares protocol requires $O(n)$ messages, where n is the number of all entities who have feedback about the target. The performance of the protocols by Gudes et al. can be better since $t \leq n$. However, a property of their protocols is that they rely on TTPs, which our protocol does not. The third protocol runs a version of the protocol by Pavlov et al. and requires at least $O(t^2)$ messages.

Androulaki et al. [21] 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. The reputation protocol has two key objectives: 1) unlinkability between pseudonyms and true identities, and 2) unforgeability, that is, users are unable to forge good reputation. These objectives are achieved primarily by using e-cash [22, 23], a cryptographic digital currency that offers anonymity and unforgeability. Reputation is awarded in the form of e-coins called *repcoins*. According to the authors, the querying of reputation requires a constant number of messages. The weakness of the system is that it requires the presence of a centralized entity called *the bank*, which implies that the system is not truly decentralized. Additionally, the system also requires that all communication take place over an anonymous network, such as a network using Onion routing [24]. The system by Androulaki et al. [21] achieves stronger privacy with fewer messages required for querying reputation than our k-shares protocol. However, our protocol is truly decentralized and does not require computationally expensive cryptography or anonymous networks to operate. Ismail et al. [25, 26] also present privacy preserving reputation protocols based on e-cash, however, they rely on TTPs, which our protocol does not.

The decentralized reputation system proposed by Kinateder and Pearson [27] 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. Voss et al. [28] and Bo et al. [29] 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. [30] avoids the hardware modules, however, it requires an anonymous routing infrastructure at the network level. Our approach does not mandate hardware modules or specialized platforms such as anonymous networks.

Our work may also be compared with some schemes for privacy preserving routing in structured Peer-to-Peer (P2P) networks based on Distributed Hash Tables (DHTs), such as a recent algorithm by Mittal et al. [31] (X-Vine). DHT-based structured P2P networks guarantee that a node can route a message to any other node in the network in a bounded number of hops (generally $O(log\ N)$ hops, where N is the total number of nodes). Privacy preserving DHT-based structured P2P networks have the additional goal that messages can be routed in the network without revealing the true identities (for example, IP addresses) of the source and the destination nodes. Mittal et al. [31] propose a privacy preserving DHT routing algorithm that relies on an overlay network of social relationships between nodes to preserve their anonymity. Similar to our protocol, a node in X-Vine depends on its trusted fellow nodes to help it preserve its privacy. In X-Vine, a node reveals its IP address only to its trusted nodes. Its true identity remains private as long as those trusted nodes are honest. In our k-shares protocol, the privacy of a node's feedback is also preserved as long as its trusted nodes are honest. However, a main difference in our protocol is that all trusted nodes must be dishonest to reveal the node's private data, whereas in the protocol by Mittal et al. only one of the trusted nodes can breach privacy by revealing the IP address of the node.

6. Future Work

6.1. Privacy under the Malicious Adversarial Model

Agents under the malicious adversarial model may deviate from the protocol as and when they deem necessary. They may attempt to learn private inputs as well as to disrupt the protocol for honest agents. Two of the most problematic actions that malicious agents may take are as follows: 1) drop messages that they are supposed to send, and 2) provide out of range values as their inputs.

We would like to extend our k-shares protocol by using cryptographic techniques for security under the malicious adversarial model. To counter the first action, we propose that messages between source agents are encrypted with an additive homomorphic public key cryptosystem and relayed through the querying agent en route to the destination source agent. The encryption would prevent the querying agent from learning the contents of the message, however, the querying agent would learn the identity of an agent if it drops an expected message. The querying agent can then exclude the identified malicious agents and restart the protocol with the remaining source agents. The issue of out of

range values can be countered as follows: each source agent provides a zero-knowledge proof of set membership (such as the one described by Baudron et al.[32]) to the querying agent demonstrating that the sum of the prepared shares lies in the correct range. Encryption of the prepared shares with an additive homomorphic cryptosystem (such as the Paillier cryptosystem [33]) and the said zero-knowledge proof would allow the querying agent to confirm that the feedback value lies in the correct range without being able to learn the value.

The extended protocol retains the novel properties of the k-shares protocol that allow an agent to rely on fellow agents that it considers trustworthy and to be able to quantify the risk to privacy before submitting feedback. We propose a detailed specification and analysis of this extended protocol as future work. The decentralized privacy preserving reputation protocol for the malicious adversarial model by Pavlov et al. requires $O(n^3)$ messages primarily due to a costly witness selection scheme. We believe that the number of messages required by our protocol would be much lower as we utilize trust between agents to achieve security instead of an expensive witness selection scheme.

6.2. Privacy in Conjunction with Solutions for other Challenges

In addition to lack of privacy, there are a number of other challenges that are faced by reputation systems. Two of these challenges are as follows: 1) *Sybil attack*: an attacker creates multiple pseudonyms in the system and may use these pseudonyms for malicious actions such as self-promotion or slandering. For example, all the fake pseudonyms could assign the highest feedback to the attacker to achieve self-promotion. Alternatively, the fake pseudonyms could assign the lowest possible feedback to an honest agent to accomplish slandering. 2) *Subjectivity*: different users submit feedback with different bias. For example, one user always submits relatively high feedback and another user always submits relatively low feedback. The feedback of the users is thus not normalized. As future work, we would like to address the issue of privacy in conjunction with other challenges.

Let's consider the issue of sybil attack. One possible solution is to require each source agent in the k-shares protocol to digitally sign the messages that it sends to the querying agent. The querying agent verifies that each source agent uses a unique private-public key pair to sign the messages. Assuming that the certificate authority is trustworthy and it issues no more than one private-public key pair to a single user, any user would not be able to submit feedback through multiple source agents without detection. We would also like to explore solutions that do not require restrictive cryptographic tools.

Let's now consider the problem of subjectivity. In a previous article [34], we proposed the following non-privacy preserving solution to the issue of subjectivity: Instead of reporting the feedback value, a source agent reports the percentile of the value in relation to all his other feedback. The querying agent looks up the value at the corresponding percentile in his own list of feedback values and considers that as the feedback submitted by the source agent. This approach normalizes the submitted feedback values to the bias of the querying

agent. We would like to update the k-shares protocol to utilize this technique in order to address the issue of subjectivity in conjunction with privacy.

7. Conclusion

In this article, we have presented the k-shares privacy preserving reputation protocol. The foundation of the protocol is a formal framework that unifies the concepts of trust, reputation, and privacy.

A defining characteristic of the protocol is that an agent $\boldsymbol{a}$ himself selects the agents that are critical for preserving its privacy. The selection is based on $\boldsymbol{a}$'s knowledge of the trustworthiness of those agents in the context of preserving privacy. The consequence is that agent $\boldsymbol{a}$ is able to quantify and maximize the probability that its privacy will be preserved. This also enables an extension that allows agents to abstain when their privacy is at risk.

Another key characteristic is that the number of agents that each agent sends shares to is limited to $k \ll n$. This results in a protocol that is very efficient and requires an exchange of only $O(n)$ messages. This design choice is validated by the experiment results, which show that the privacy of a high majority of agents can be assured with k as small as 2. It is also observed that increasing k to values approaching $n - 1$ provides no significant advantage.

Acknowledgment

This work was supported in part by the French Agence Nationale de la Recherche (ANR) under Grant ANR-10-SEGI-013 (SocEDA).

References

[1] CyberSource Corporation, Cybersource 12th annual online fraud report, http://www.cybersource.com/ (2011).

[2] Aladdin Knowledge Systems Ltd., Attack intelligence research center annual threat report, http://www.aladdin.com/pdf/airc/AIRC-Annual-Threat-Report2008.pdf (2008).

[3] Unvarnished, Unvarnished – community-contributed reviews for business professionals, http://www.getunvarnished.com/ (July 2010).

[4] Duedil, Duedil – transparent, constructive feedback on your profile, http://www.duedil.com/ (July 2010).

[5] C. Costa, 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, 2007.

[6] L. Yu, A reputation system for bittorrent peer-to-peer file-sharing networks, Master's thesis, University of Wollongong, Australia (2006).

[7] S. D. Kamvar, M. T. Schlosser, H. GarciaMolina, The eigentrust algorithm for reputation management in p2p networks, in: Proc. of the 12th Intl. Conf. on World Wide Web (WWW 2003), 2003.

[8] J. Hu, 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.

[9] S. Buchegger, 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, 2004.

[10] S. Buchegger, J.-Y. L. Boudec, Performance analysis of the confidant protocol, in: Proc. of the Third ACM Intl. Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC'02), Lausanne, Switzerland, 2002.

[11] P. Resnick, R. Zeckhauser, Trust among strangers in internet transactions, Advances in Applied Microeconomics (11) (2002) 127–157.

[12] D. Gambetta, Trust: Making and Breaking Cooperative Relations, Dept. of Sociology, U. of Oxford, 2000, Ch. Can We Trust Trust?, pp. 213 – 237.

[13] O. Goldreich, The Foundations of Crypto. - Vol. 2, Cambridge Univ., 2004.

[14] E. Pavlov, J. S. Rosenschein, Z. Topol, Supporting privacy in decentralized additive reputation systems, in: Proc. of the 2nd Intl. Conf. on Trust Management (iTrust 2004), 2004.

[15] O. Hasan, E. Bertino, L. Brunie, Efficient privacy preserving reputation protocols inspired by secure sum, in: Proceedings of the 8th International Conference on Privacy, Security and Trust (PST 2010), Ottawa, Canada, 2010.

[16] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, M. Y. Zhu, Tools for privacy preserving distributed data mining, SIGKDD Explorations.

[17] F. Kerschbaum, A verifiable, centralized, coercion-free reputation system, in: Proceedings of the 8th ACM workshop on Privacy in the electronic society (WPES'09), ACM, New York, NY, USA, 2009.

[18] O. Hasan, Privacy preserving reputation systems for decentralized environments, Ph.D. thesis, INSA Lyon, France (2010).

[19] E. Gudes, N. Gal-Oz, A. Grubshtein, Methods for computing trust and reputation while preserving privacy, in: Proc. of the 23rd Annual IFIP WG 11.3 Working Conference on Data and Applications Security, 2009.

[20] N. Gal-Oz, E. Gudes, 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.

[21] E. Androulaki, S. G. Choi, S. M. Bellovin, T. Malkin, Reputation systems for anonymous networks, in: Proc. of the 8th Privacy Enhancing Technologies Symp. (PETS 2008), 2008.

[22] D. Chaum, Blind signatures for untraceable payments, in: Proc. Advances in Cryptology (CRYPTO '82), 1982.

[23] D. Chaum, Blind signature systems, in: Advances in Cryptology (CRYPTO'83), 1983.

[24] R. Dingledine, N. Mathewson, P. F. Syverson, Tor: The second-generation onion router, in: Proceedings of the USENIX Security Symposium, 2004.

[25] R. Ismail, C. Boyd, A. Josang, S. Russell, Strong privacy in reputation systems, in: Proc. of the 4th Intl. Workshop on Info. Security Apps. (WISA'03), 2004.

[26] R. Ismail, C. Boyd, A. Josang, S. Russell, Private reputation schemes for p2p systems, in: Proc. of the 2nd Intl. Workshop on Security in Info. Systems, 2004.

[27] M. Kinateder, S. Pearson, A privacy-enhanced p2p reputation system, in: Proc. of the 4th Intl. Conf. on E-Commerce and Web Techs., 2003.

[28] M. Voss, A. Heinemann, M. Muhlhauser, A privacy preserving rep. system for mobile info. dissemination networks, in: SECURECOMM, 2005.

[29] Y. Bo, Z. Min, L. Guohuan, A reputation system with privacy and incentive, in: Proc. of the 8th ACIS Intl. Conf. on Soft. Eng., AI, Networking, and Parallel/Distributed Comp. (SNPD'07), 2007.

[30] M. Kinateder, R. Terdic, K. Rothermel, Strong pseudonymous comm. for p2p rep. systems, in: Proc. of the ACM Symp. on Applied Comp., 2005.

[31] P. Mittal, M. Caesar, N. Borisov, X-Vine: Secure and pseudonymous routing in DHTs using social networks, Tech. rep., Dept. of Electrical and Computer Eng., Univ. of Illinois at Urbana Champaign (May 2011).
URL `https://netfiles.uiuc.edu/mittal2/www/x-vine.pdf`

[32] O. Baudron, P.-A. Fouque, D. Pointcheval, J. Stern, G. Poupard, Practical multi-candidate election system, in: Proceedings of the Twentieth Annual ACM Symposium on Principles of Distributed Computing, 2001.

[33] 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.

[34] O. Hasan, L. Brunie, J.-M. Pierson, E. Bertino, Elimination of subjectivity from trust recommendation, in: Proceedings of the 3rd IFIP International Conference on Trust Management (TM 2009), 2009.

Protocol: Semi-Honest-k-Shares
Participants: Agents: q, t, $S_t \equiv S_{t,\psi} = \{a_1 \ldots a_n\}$. Agents q, t, and a subset of $S_{t,\psi}$ of size $m < n$ are dishonest, however, q wishes to learn the correct output.
Input: Each source agent 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 context ψ.
Setup: Each agent a maintains $S_a \equiv S_{a,\psi}$, the set of its source agents in context ψ. $k \ll n$ is a constant.

Events and Associated Actions (for an Agent a):

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

▷ initiate query
send tuple $(\text{REQUEST\_FOR\_SOURCES}, \psi)$ to t
receive tuple $(\text{SOURCES}, \psi, S_t)$ from t
for each agent $v \in S_t$
    do $J_v \leftarrow \phi$
$S'_t \leftarrow S_t$
$r \leftarrow 0$
$q \leftarrow a$
$s \leftarrow$ timestamp()
send tuple $(\text{PREP}, q, t, s, S_t)$ to each agent $v \in S_t$

tuple $(\text{REQUEST\_FOR\_SOURCES}, \psi)$ **received from agent** q

send tuple $(\text{SOURCES}, \psi, S_a)$ to q

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

$I \leftarrow \phi$
$J \leftarrow \phi$
$\sigma_a \leftarrow 0$
$U_a \leftarrow$ set_of_trustworthy$(a, S_t - a)$
$k_a \leftarrow |U_a|$
for $i \leftarrow 1$ **to** k_a
    do $x_{a,i} \leftarrow \text{random}(-X, X)$
$x_{a,k_a+1} \leftarrow l_{at} - \sum_{i=1}^{k_a} x_{a,i}$
send tuple $(\text{RECIPIENTS}, q, t, s, U_a)$ to agent q
for each agent $u_{a,i} \in U_a = \{u_{a,1} \ldots u_{a,k_a}\}$
    do send tuple $(\text{SHARE}, q, t, s, x_{a,i})$ to agent $u_{a,i}$

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

for each agent $u \in U_v$
    do $J_u \leftarrow J_u \cup v$
$S'_t \leftarrow S'_t - v$
if $S'_t = \phi$
  then $S'_t \leftarrow S_t$
      for each agent $w \in S_t$
          do send tuple $(\text{SENDERS}, q, t, s, J_w)$ to agent w

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

$I \leftarrow I \cup v$
$\sigma_a \leftarrow \sigma_a + x_v$
if $I = J$
  then $\sigma_a \leftarrow \sigma_a + x_{a,k_a+1}$
      send tuple $(\text{SUM}, q, t, s, \sigma_a)$ to agent q

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

$J \leftarrow J_a$
if $I = J$
  then $\sigma_a \leftarrow \sigma_a + x_{a,k_a+1}$
      send tuple $(\text{SUM}, q, t, s, \sigma_a)$ to agent q

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

$S'_t \leftarrow S'_t - v$
$r \leftarrow r + \sigma_v$
if $S'_t = \phi$
  then $r_{t,\psi} \leftarrow r/n$

Figure 1: Protocol: Semi-Honest-k-Shares