Schemes for privately computing trust and reputation

Nurit Gal-Oz, Niv Gilboa and Ehud Gudes

Dept. of Computer Science and Deutsche Telekom Laboratories
at Ben-Gurion University
Beer-Sheva, 84105
Israel
e-mail:galoz@cs.bgu.ac.il, gilboan@bgu.ac.il, ehud@cs.bgu.ac.il

Abstract. Trust and Reputation systems in distributed environments attain widespread interest as online communities are becoming an inherent part of the daily routine of Internet users. Several models for Trust and Reputation have been suggested recently, among them the Knots model [8]. The Knots model provides a member of a community with a method to compute the reputation of other community members. Reputation in this model is subjective and tailored to the taste and choices of the computing member and those members that have similar views, i.e. the computing member's *Trust-Set*. A discussion on privately computing trust in the Knots model appears in [16]. The present paper extends and improves [16] by presenting three efficient and private protocols to compute trust in trust based reputation systems that use any trust-sets based model. The protocols in the paper are rigorously proved to be private against a semi-honest adversary given standard assumptions on the existence of an homomorphic, semantically secure, public key encryption system. The protocols are analyzed and compared in terms of their privacy characteristics and communication complexity.

1 Introduction

Recent years have seen a substantial growth of virtual communities across the Internet. These enable people to gather around some common goal or shared interest. The accessibility of information and services offered by these communities, makes it both possible and legitimate to communicate with strangers and carry out interactions anonymously, as rarely done in "real" life. On the other hand, virtual communities are prone to many types of deception - possibly exposing users to various threats - ranging from people forging their identity and imposing as others, to people giving extremely bad or extremely good ratings to other members unrelated to the service they have received from them. Trust and Reputation systems provide communities with means to reduce the potential risk when communicating with people hiding behind virtual identities. These systems utilize the experience and knowledge accumulated and shared by all participants for assigning reputation values to individuals. Moreover, they attempt to identify dishonest members and prevent their negative effect.

Centralized reputation systems, such as the commercial system eBay [1] , collect and store reputation ratings from feedback providers in a centralized reputation database. In eBay, for example, both buyers and sellers participating in a transaction, may provide one of three possible feedbacks: positive (+1), neutral (0), and negative (-1). The reputation score of a user is simply the sum of her accumulated ratings over a period of six months.

Several authors have noted that reputation is a much more complex concept than simply aggregation of ratings. It may depend on interaction of multiple attributes (also exist in eBay), on the certainty of the rating [18], on the time the interaction and rating was performed, and on the trust between members. The last factor, trust between members, is crucial in obtaining reputation which is specifically compatible with a user profile or preferences. One usually gives much higher weight to ratings provided by people she has trust in. Trust between members is considered among others by [5], and when anonymity of users is required, trust between members may be computed based on the similarity of their past ratings. (see [8]).

However, in the above systems and models the reputation engine assumes knowledge of all ratings and other reputation factors, and does the computation centrally. As a result, the raters suffer a severe loss of privacy. An empirical study conducted by [27] on data sets extracted from eBay's reputation system reported a high correlation between buyer and seller ratings. Moreover, most of the feedback provided was positive. A possible explanation for these results is that when feedback providers' identities (or pseudo-identities) are known, reputation ratings are provided based on reasons of reciprocation and retaliation, not properly reflecting the trustworthiness of the rated parties. Thus preserving privacy while computing reputation becomes an important issue.

Decentralized reputation systems, on the other hand, do not make use of a central repository to collect and report reputation ratings [32]. In these types of systems, both the reputation of users and the ratings they have given may be stored locally and known only to the corresponding user. The challenge in these system it to compute the reputation without revealing the private data. Its important to emphasize that depending on the reputation model, there may be different data which may be considered private, such as the rating itself, the weight assigned to any specific rating, the identity of the raters, the trust between members, etc.

A recent paper on this topic [24] suggested several privacy preserving schemes for computing reputation when the reputation computation is very simple (similar to eBay). That is, the reputation is based on the summation of reputation scores.

The current paper advances the state of the art as it uses a more advanced model of reputation computation which considers the trust members have in one another. We use some notations of the *Knots* model [8] to demonstrate and formulate our schemes however it is important to note that the ideas presented may apply to any model in which trust between members is an important factor in computing reputation. Specifically we refer to trust based reputation mod-

els that use the feedback of a set of trusted members (trust-set) to compute reputation. We present *three* different methods for computing trust and reputation privately and discuss their comparative advantages and disadvantages. Each method offers a slightly different degree of privacy and communication overhead.

We use the terms *Trust* and *Reputation*, as introduced by [19]. Trust is a subjective expectation a member has about another member's future behavior, based on the history of their encounters, while Reputation is the perception that an agent creates through past actions about its intentions and norms, and is computed by some aggregation of members ratings of previous interactions. Normally, the trust we have in an unknown person is based on their reputation. The exact use of these two terms in our paper will become clearer in Section 3.

The rest of the paper is organized as follows. Section 2 provides an overview of the related work. In section 3, we formally define our trust-set-aware model which is essential for understanding the rest of the paper, while in Section 4 we formally define secure computation in our context. Section 5 describes the three different schemes and in Section 6 we conclude and give some future research directions.

2 Background and Related Work

A community (virtual or online community) is a group of entities (e.g. people, nodes, peers or agents acting on behalf of people), interacting via computer networks for a common interest. Different communities serve various needs of social groups through different levels of interactions. A community of strangers is a community of anonymous entities who would like to participate, i.e. contribute to and benefit from the community activities, without revealing personal information. This is in contrast to the recently popular Internet communities of identified users (e.g. 'Facebook', 'LinkedIn' etc.)

A review on trust and reputation systems is provided by Josang et. al in [2]. Their review discusses the semantics of the trust and reputation concepts and the relations between them. The authors also provide an overview of reputation computation models and existing applications of online reputation systems. Sabater et al [28] also present an overview of several computational trust and reputation models that have been implemented and classify them according to several criteria, such as the source of the information used, the assumptions made on agents behavior, the visibility of trust, and whether reputation is considered as a personal/subjective property or as a global property. The concern for privacy in communities in general, and the privacy of reputation information in particular, was discussed in several papers [19, 17, 30, 26, 7] as explained next.

In [19] the authors discuss the issues of privacy in a P2P network where the reputation information is distributed among the P2P nodes. The following aspects are analyzed: How the requirements for fair use practices reflect on the system design? What classes of information may be leaked? How to manage the risks related to social and technical issues? However, a specific method of computing reputation is not discussed. In [17] a distributed trust management

system which is constructed from a two level hierarchy is described. The high level is composed of brokers which are responsible for aggregating trust information from their individual local nodes. This provides some privacy from one broker to another, although no privacy is provided at a single broker's network.

Steinbrecher in [30] presents an information theoretic model of reputation privacy. She tries to model the amount of privacy lost when a single user uses different pseudonyms in the same community or in different communities, or when a user change his/her pseudonym. Her measure enables the estimate of unlinkability provided by such a pseudonym change. In [26] the authors discuss the issue of privacy when a single user is a member in multiple communities and requires the transfer of the reputation between these communities creating the concept of cross-community reputation. Cross community reputation requirements were analyzed in [7] including the issues of privacy, user control vs. community control, ontology matching and others.

The closest paper to the present work and the one in [16] is the paper by Pavlov et. al [24] for privately computing reputation information, when the reputation is defined as an additive reputation system (e.g. the Beta reputation system by Josang [18]). The authors present three algorithms for computing additive reputation information with various degrees of privacy and with different abilities for protecting against malicious users. The paper starts with a method for "witness selection" which reduces the risk of selecting dishonest witnesses. Then, a simple scheme is presented which is very efficient but is vulnerable to collusion of even two witnesses. The second scheme is more resilient towards curious users although still vulnerable to collusions and uses a secret splitting scheme. The last method provides the most secure protocol which uses the verifiable secret sharing scheme [25] which is based on Shamir's secret sharing scheme [29]. The complexity of the scheme is quite high since it requires $O(n^3)$ messages where n is the number of contributing nodes. Another relevant paper was presented recently by Nin et. al [21]. [21] also uses Homomorphic encryption similar to two of our schemes, but their reputation model is quite different and is not a Trust-set based model like ours.

Finally, in our previous paper [16] we described three methods for computing reputation privately, where the last one uses the scheme of [24]. The contributions of the present paper over [16] are in two areas:

1. Out of the three presented protocols, two are completely new and use the idea of Homomorphic encryption [23]. In addition, some of the strong assumptions made in [16] are removed here.
2. We present a formal framework for the three protocols and define precisely the concept of privacy in our context. Then, using this framework we prove rigorously the correctness of the three protocols.

3 Computing reputation based on Trust sets

The *Trust-Set* of a member at some point in time is the subset of community members she trusts above some level. The idea behind the trust-set concept is to

allow every member to rely on members that provided her with accurate feedback in the past. The benefit of using trust-sets is in limiting the recommendation process to a smaller group of members that are better qualified for the task, while increasing the chance of getting more accurate results. While many trust based reputation systems (e.g. [31, 33]) use trust-sets, they differ in two major aspects: the criteria they use to generate these sets and the method for calculating trust in each member of the trust-set. These aspects have no affect on the privacy preserving schemes we present and therefore they are out of the scope of this paper.

The Knots model is a trust-based reputation model introduced in a previous work [8] for large-scale virtual communities. It is designed for communities in which members typically do not reveal their real identities. We consider for example communities in which experts in specific fields offer their advice and consulting services to community members seeking such services. The Knot model uses trust-sets as well as Knots, a relaxed form of trust-set. Knots are designed among other things to cope with the problem of sparsity i.e. insufficient number of trust relations between members in the community. Without loss of generality we adopt the notation form the knot-aware model, however the underlying approach of using trust relations among members as a weight when computing reputation exists in other models as well:

- *TrustMember* (TM) is trust in the context of recommendations. More specifically, it is a trust value that quantifies the extent by which one member relies on another member to rate experts "correctly".
- *TrustExpert* (TE) is trust in the context of experts. More specifically, it is a trust value that quantifies the extent by which a member relies on an expert to successfully provide the service it requires.
- An α ***- Trust Set*** of a member A, is the set of all members whom A trusts with level α or more.

The motivation for our current paper is to provide trust based reputation models (such as the knot model), with a means to compute reputation in a distributed manner while preserving private information. In this context the trust one member has in another (TM) and the trust a member has in an expert (TE), are considered private information. One would prefer not to reveal her trust in a member from which she gets recommendations since this trust value is used relative importance. On the other hand the recommending party may prefer to keep her recommendation private due to the fear of retaliation. Revealing these values to malicious parties exposes the community to the risk of manipulating recommendations.

The problem of privacy we address can be described as follows. Member A needs to compute its trust in an expert x, based on the experience other members of the community have had with this expert. Using the terms introduced in [8], this trust can vary from using the concept of *Trust Expert* based only on A's own trust-set, to using the expert's local reputation based on the Knot to which A belongs to, to using the expert's global reputation based on the entire community. We focus on Trust-set based computation since the other options

are quite similar. The trust of A in x using the experience of her Trust-set can be computed according to:

$$TE(A,x) = \frac{\sum_{\substack{B_i \in TrustSet(A), \\ DTE(B_i,x) \neq \perp}} DTE(B_i,x) \cdot TM(A,B_i)}{\sum_{\substack{B_i \in TrustSet(A), \\ DTE(B_i,x) \neq \perp}} TM(A,B_i)}$$

where:

- $DTE(B_i,x)$ - the trust member B_i has in expert x based on her own accumulated experience.
- $TM(A,B_i)$ - the trust member A has in member B_i.

We assume that $TM(A,B)$ is known to agent A since it reflects her private information. Therefore the denominator in this formula is easy to compute by A without disclosing private information. The nominator is a sum of products of two terms, the first one is assumed to be known only to the individual agents B_i and the second one is known to A. Therefore the challenge we face is to privately compute the following sum of products, denoted by $\rho(A,x)$:

$$\rho(A,x) = \sum_{B_i=1}^{|S|} DTE(B_i,x) \cdot TM(A,B_i)$$

where S denotes the trust-set of A.

Each product in this sum expresses the indirect trust that member A has in expert x based on B_i's direct trust in x, denoted

$$ITM(A,B_i,x) = TM(A,B_i) \cdot DTE(B_i,x)$$

We use this notation in Section 5 to explain our schemes.

4 Formal framework

We now define the exact notion of secure computation in our context. Members of a trust set wish to jointly compute a function of their private inputs so that one of these members obtains an output, while no member obtains information on the input of other members. We assume that the participants in our protocols are semi-honest. In other words, each player executes the protocol as required, but may attempt to learn information on the input of other parties. Thus, a computation is secure if it is *private* for each of the parties, leaking no unnecessary information about a party's input.

Let Π be a protocol for k parties to compute a function g. The input of the i-th party is denoted x_i and the protocol output $g(x_1,\ldots,x_k)$ is obtained by the first party, denoted by A. An adversary controls a set I of parties and receives the "view" of every party in I. The view of a party includes its input, output (if any) and all intermediate messages that it receives. Informally, we say that a protocol is private if for every adversary there is a probabilistic, polynomial time algorithm with access only to the adversary's input and output, that simulates

the view of the adversary. Intuitively, if the view of an adversary can be simulated from this adversary's input and output, then the adversary learns nothing about the input of other parties from an execution of Π.

Our formal definitions follow the full framework set in [10] and [11]. We use only those elements of the definition framework that are necessary in our setting.

Definition 1. *A function $\mu : \mathbb{N} \longrightarrow \mathbb{N}$ is* negligible *if for every polynomial $p : \mathbb{N} \longrightarrow [0,1]$, there exists N such that $\mu(n) < \frac{1}{p(n)}$ for every $n > N$.*

Two distribution ensembles are computationally indistinguishable if no efficient algorithm can decide with good probability, whether its input is chosen according to the first distribution or the second distribution. We regard a distribution ensemble as a collection of distributions that are indexed by two parameters: a binary string a and a security parameter n represented in unary form. In our setting, a is the input for a protocol, the distribution is over all the messages of a protocol (for a subset of parties) and is induced by coin tosses of each party as it executes its part of the protocol. The security parameter 1^n determines the required length of cryptographic keys to ensure privacy of the protocol. Formally:

Definition 2. *Let $X = \{X(a,1^n)\}_{n\in\mathbb{N},a\in\{0,1\}^*}$ and $Y = \{Y(a,1^n)\}_{n\in\mathbb{N},a\in\{0,1\}^*}$ be two distribution ensembles. We say that X and Y are* computationally indistinguishable *if for every probabilistic, polynomial time algorithm D , there exists a negligible function μ such that for every $a \in \{0,1\}^*$:*

$$|Pr[D(X(a,1^n)) = 1] - Pr[D(Y(a,1^n)) = 1]| < \mu(n).$$

We denote computational indistinguishability of two ensembles by $X \stackrel{c}{\equiv} Y$.

We restrict an adversary to corrupting a subset of parties *statically*. In other words, the adversary controls the same subset of parties throughout the execution of a protocol. The adversary may corrupt only specific subsets of parties.

Definition 3. *Let Π be a communication protocol for k parties to compute a function g. Let $2^{\{1,\ldots,k\}}$ be the set of all subsets of parties in the protocol. We say that U is the* adversary structure *if $U \subseteq 2^{\{1,\ldots,k\}}$ includes exactly all subsets of parties, I, such that the adversary can corrupt all parties in I simultaneously.*

We denote by $\text{view}_I^{\Pi}(X_I, 1^n)$ the aggregated input, output and protocol messages of all parties in $I = \{i_1, \ldots, i_\ell\}$ as they execute a protocol Π with security parameter 1^n on input $X_I = (x_{i_1}, \ldots, x_{i_\ell})$. Next, we formally define multi-party private computation.

Definition 4. *Let $g : (\{0,1\}^*)^k \longrightarrow \{0,1\}^*$ be a function mapping k binary strings to a single binary string. Let Π be a communication protocol for k parties, such that if the i-th party has input x_i for $i = 1, \ldots, k$ then after the parties execute Π, the first party has output $g(x_1, \ldots, x_k)$. Let U be an adversary structure. For every subset of parties I, such that $I = \{i_1, \ldots, i_\ell\}$, denote by X_I the*

input vector $(x_{i_1}, \ldots, x_{i_\ell})$ *and by* O_I *the output obtained by members of* I. Π privately *computes* g *if for every* I *such that* $I \in U$, *there exists a probabilistic, polynomial time algorithm* S *such that when* n *ranges over* $\mathbb{N}$ *and* X_I *ranges over* $(\{0,1\}^*)^{|I|}$:

$$\{S(X_I, O_I, 1^n)\} \stackrel{c}{\equiv} \left\{view_I^{\Pi}(X_I, 1^n)\right\}$$

We use semantically-secure public-key encryption [14] to construct our protocols is . Semantic security means that a ciphertext does not leak any information about the plaintext when the private key is unknown. Furthermore, even if an adversary has some a-priori knowledge about the plaintext (e.g. a possible range of values for the plaintext) it cannot infer additional information from the ciphertext. Let $f(\cdot)$ be information on the plaintext that the adversary is trying to obtain and let $h(\cdot)$ be information on the plaintext that the adversary already knows.

Let an encryption scheme be three algorithms: a key generation algorithm G, an encryption algorithm E and a decryption algorithm D. On input 1^n, G outputs a randomly chosen key of appropriate length. Intuitively, an algorithm that has an encryption $E(X)$ and $h(X)$ does not have significant advantage computing $f(X)$ compared to an algorithm that has only $h(X)$. The formal definition follows [11]:

Definition 5. *Let* (G, E, D) *be a public key encryption scheme.* (G, E, D) *is semantically-secure if for any probabilistic, polynomial time algorithm* P *there exists a probabilistic, polynomial time algorithm* P' *such that for probability ensemble* $\{X_n\}_{n \in \mathbb{N}}$, *with* $|X_n|$ *polynomial in* n, *and two polynomially bounded functions* f, h, *there exists a negligible function* μ *such that for every* n

$$\left| Pr[P_{X_n}(E, h) = f(X_n)] - Pr[P'_{X_n}(h) = f(X_n)] \right| < \mu(n)$$

where $P_{X_n}(E, h) \stackrel{\triangle}{=} P(1^n, E(X_n), h(X_n))$ *and* $P'_{X_n}(h) \stackrel{\triangle}{=} P'(1^n, 1^{|X_n|}, h(X_n)) = f(X_n)$

Intuitively, semantic security enables one party in a multi-party computation to send encrypted data to another party without compromising privacy. A simulator for the receiving party can simulate this encrypted data by encrypting some data that it generates. Semantic security ensures that real encrypted data can't be distinguished from simulated encrypted data. This intuition is formalized in Lemma 1. Due to space limitations we leave out the proof of this lemma.

Lemma 1. *Let* (G, E, D) *be a semantically secure public key encryption scheme, let* n *be a security parameter and let* $x_1, \ldots, x_\ell$ *be chosen according to a distribution* D *that can be efficiently constructed. Let* Π *be a polynomial time (in* n*) multi-party protocol to compute a function* g *and let* I *be a set of parties, such that* $view_I^{\Pi}(X_I, 1^n)$ *is comprised of* X_I, O_I *and* $E(x_1), \ldots, E(x_\ell)$. $G(1^n)$ *generates a public and private key pair. Members of* I *know the public key, but not the private key. Then* Π *privately computes* g *for an adversary structure* U, $U = \{I\}$.

5 The Protocols

In this section we present three different schemes for privately computing reputation and we prove the correctness of these schemes based on the above framework under the assumption of semi-honest participants. In all three schemes we compute the reputation of x in the eyes of A $\rho(A, x)$, using input from A's trust-set S (see section 3).

5.1 Scheme 1: An external party

In this scheme, A learns only $\rho(A, x) = \sum_{B_i=1}^{|S|} DTE(B_i, x) \cdot TM(A, B_i)$ (see section 3), while B_i receives no information at all, for every i. We assume the existence of an additional party Z that must not collude with A.

The main tool we use in this scheme is public-key, homomorphic encryption. In such encryption there is a modulus m and an efficiently computable function ϕ that maps a pair of encrypted values $(E_K(x), E_K(y))$, where $0 \leq x, y \leq m$, to a single encrypted element $\phi(E_K(x), E_K(y)) = E_K(x + y \bmod m)$. In many homomorphic encryption systems the function ϕ is multiplication modulo some integer N. Given a natural number c and an encryption $E_K(x)$, it is possible to compute $E_K(c \cdot x \bmod m)$, without knowing the private key [1].

There are quite a few examples of homomorphic encryption schemes known in the cryptographic literature, including [14, 3, 20, 22] and [23]. There are also systems that allow both addition and multiplication of two encrypted plaintexts, e.g. [4] (only a single multiplication is possible for a pair of ciphertexts) and [9]. All of these examples of homomorphic cryptosystems are currently assumed to be semantically secure.

We assume that all participants in the protocol know prior to the beginning of the protocol a semantically-secure, public key and homomorphic encryption scheme (G, E, D), an associated function ϕ and a security parameter n. We can assume that the modulus m of the homomorphism is large enough, that is $m > \rho(A, x)$, since in many of the well-known homomorphic cryptosystems the key can be chosen to accommodate an arbitrarily large m. We require that for every i, $TM(A, B_i)$ and $DTE(B_i, x)$ have integral values.

The total computational complexity of this scheme is $O(|S|)$ encryptions, decryptions and multiplications. The total communication complexity is $O(|S|)$ times the size of a ciphertext.

Proposition 1. *Let $A, Z, B_1, \ldots, B_{|S|}$ be the parties, let the input of A be $TM(A, B_1), \ldots, TM(A, B_{|S|})$ and for all $i = 1, \ldots, |S|$ let the input of B_i be $DTE(B_i, x)$. Let g be a function that maps the ordered sequence of pairs $\langle TM(A, B_i), DTE(B_i, x)\rangle_{i=1,\ldots,|S|}$ to the value $\rho(A, x) = \sum_{B_i \in S} ITM(A, B_i, x)$.*

[1] Set $\beta = E_K(0)$ and let the binary representation of c be $c = c_k c_{k-1} \ldots c_0$. Go over the bits $c_k, \ldots, c_0$ in descending order. If $c_j = 0$ set $\beta = \phi(\beta, \beta)$ and if $c_j = 1$ set $\beta = \phi(\phi(\beta, \beta), E_K(x))$. If ϕ is modular multiplication, this algorithm is identical to regular modular exponentiation.

Algorithm 1 Computing a scalar product

1: A runs $G(1^n)$ and obtains a private-public key pair.
2: A sends to B_i the ciphertext $E_{K_A}(TM(A, B_i) \bmod m)$, for every $B_i \in S$.
3: **for all** $B_i \in S$ **do**
4: B_i uses $E_{K_A}(TM(A, B_i) \bmod m)$ and $DTE(B_i, x)$ to compute $E_{K_A}(ITM(A, B_i, x) \bmod m)$.
5: B_i sends $E_{K_A}(ITM(A, B_i, x) \bmod m)$ to Z.
6: Z uses ϕ to compute $E_{K_A}(\sum_{B_i \in S} ITM(A, B_i, x) \bmod m)$ and sends this encrypted value to A.
7: A obtains $\rho(A, x) = \sum_{B_i \in S} ITM(A, B_i, x)$ by decryption.

Let the adversary structure U include all subsets I of $\{A, Z, B_1, \ldots, B_{|S|}\}$ s.t. $|I \bigcap \{A, Z\}| \leq 1$. Then, the protocol in Algorithm 1 privately computes g.

Proof. If the adversary controls $I = \{A, B_{i_1}, \ldots, B_{i_\ell}\}$, then the view of the adversary is identical to its input and output, hence simulation is trivial.

If the adversary controls $I = \{B_{i_1}, \ldots, B_{i_\ell}\}$ (without A), then its view includes aside from its input, the elements $\{E_{K_A}(TM(A, B_{i_j}) \bmod m)\}_{j=1,\ldots,\ell}$. By Lemma 1, Π privately computes g with respect to I.

If the adversary controls $I = \{Z, B_{i_1}, \ldots, B_{i_\ell}\}$, then its view includes aside from its input, the encrypted elements $\{E_{K_A}(ITM(A, B_{i_j}, x) \bmod m)\}_{j=1,\ldots,\ell}$. Again, Lemma 1 shows that Π privately computes g with respect to I. □

5.2 Scheme 2: No outside help

In this scheme, we dispense with the additional party Z. A learns only $\rho(A, x) = \sum_{\forall B \in S} ITM(A, B, x)$, while for every i, B_i receives no information at all. A protocol without Z is obviously desirable for those situations in which such a semi-trusted party is not available. The disadvantage of the current protocol is its greater communication complexity compared to Algorithm 1.

As is often the case in distributed, cryptographic protocols, the parties use secret sharing to distribute the input [29]. The number of distributed shares, and thus the communication complexity depends on the possible number of colluding adversarial parties. We denote this threshold by t and assume that it is part of the protocol's input.

We assume that all participants in the protocol know a semantically-secure, public key and homomorphic encryption scheme (G, E, D), an associated function ϕ and a security parameter n prior to the beginning of the protocol. We assume that the modulus m of the homomorphism satisfies the inequality $m > \rho(A, x)$.

Using algorithm 2, A computes the correct result because:

$$\begin{array}{l} \sum_{j=1}^{t} \sum_{i=1}^{|S|} TM(A, B_i) \cdot r_j^i + q_j^i \bmod m = \\ \sum_{i=1}^{|S|} TM(A, B_i) \cdot (\sum_{j=1}^{t} r_j^i) + \sum_{j=1}^{t} q_j^i \bmod m = \\ \sum_{i=1}^{|S|} TM(A, B_i) \cdot DTE(B_i, x) \bmod m = \\ \rho(A, x) \end{array}$$

Algorithm 2 No outside help

1: A runs $G(1^n)$ and obtains a private-public key pair.
2: A sends to B_i the ciphertext $E_{K_A}(TM(A,B_i) \bmod m)$, for every $B_i \in S$.
3: **for all** $B_i \in S$ **do**
4: B_i chooses $r_1^i, r_2^i, \ldots, r_t^i$ and $q_1^i, q_2^i, \ldots, q_t^i$ uniformly at random from $\{0,\ldots,m-1\}$ so that $\sum_{j=1}^t r_j^i \equiv DTE(B_i,x) \bmod m$ and $\sum_{j=1}^t q_j^i \equiv 0 \bmod m$.
5: **for all** $B_i \in S$ **do**
6: B_i computes the values: $E_{K_A}(TM(A,B_i)\cdot r_j^i + q_j^i \bmod m)$, for $j=1,\ldots,t$.
7: **for all** $j=1,\ldots,t$ **do**
8: B_i sends $E_{K_A}(TM(A,B_i)\cdot r_j^i + q_j^i \bmod m)$ to B_j.
9: **for all** $j=1,\ldots,t$ **do**
10: B_j uses ϕ on $E_{K_A}(TM(A,B_1)\cdot r_j^1 + q_j^1 \bmod m)$, $\ldots$, $E_{K_A}(TM(A,B_{|S|})\cdot r_j^{|S|} + q_j^{|S|} \bmod m)$ to compute $E_{K_A}(\sum_{i=1}^{|S|} TM(A,B_i)\cdot r_j^i + q_j^i \bmod m)$. B_j sends this value to A.
11: A decrypts and sums up all the values to obtain $\rho(A,x) = \sum_{j=1}^t \sum_{i=1}^{|S|} TM(A,B_i)\cdot r_j^i + q_j^i \bmod m$.

Denoting the maximum length of a ciphertext in this protocol by $|E(\cdot)|$, the communication complexity of steps 1-4 is $O(|S|\cdot|E(\cdot)|$, the communication complexity of step 5 is $O(t\,|S|\cdot|E(\cdot)|)$ and the communication complexity of steps $6,7$ is $O(t\cdot|E(\cdot)|$. Hence the total communication complexity of the protocol is $O(t\,|S|\cdot|E(\cdot)|)$.

Proposition 2. *Let $A, B_1, \ldots, B_{|S|}$ be the parties, let the input of A be $TM(A,B_1),\ldots,TM(A,B_{|S|})$ and for all $i=1,\ldots,|S|$ let the input of B_i be $DTE(B_i,x)$. Let g be a function that maps the ordered sequence of pairs $\langle TM(A,B_i), DTE(B_i,x)\rangle_{i=1,\ldots,|S|}$ to the value $\rho(A,x) = \sum_{B_i\in S} ITM(A,B_i,x)$. Let the adversary structure U include all subsets I of $A, B_1,\ldots,B_{|S|}$ such that $|I| \le t$. Then, the protocol in Algorithm 2 privately computes g.*

Proof. If the adversary controls $I = \{B_{i_1},\ldots,B_{i_\ell}\}$ then its view includes aside from its input, the following:

- $E_{K_A}(TM(A,B_{i_1}) \bmod m)$, $\ldots$,$E_{K_A}(TM(A,B_{i_\ell}) \bmod m)$.
- For every B_j such that $j \in \{i_1,\ldots,i_\ell\} \bigcap \{1,\ldots,t\}$, a sequence of encrypted elements: $E_{K_A}(TM(A,B_1)\cdot r_j^1 \bmod m) + q_j^1$, $\ldots$, $E_{K_A}(TM(A,B_{|S|})\cdot r_j^{|S|} + q_j^{|S|} \bmod m)$.

A simulator chooses all the necessary values to simulate $TM(A,B_i)$ and $DTE(B_i,x)$ from an appropriate distribution. The simulator can then encrypt all these $TM(A,B_i)$ and choose uniformly at random $r_1^i,\ldots,r_t^i$ and $q_1^i,\ldots,q_t^i$ from $\{0,\ldots,m-1\}$ so that $\sum_{j=1}^t r_j^i \equiv DTE(B_i,x) \bmod m$ and $\sum_{j=1}^t q_j^i \equiv 0 \bmod m$. The simulator can then use the homomorphic properties of E to compute $E_{K_A}(TM(A,B_1)\cdot r_j^1 \bmod m) + q_j^1$, $\ldots$, $E_{K_A}(TM(A,B_{|S|})\cdot r_j^{|S|} + q_j^{|S|} \bmod m)$. According to Lemma 1 the result is indistinguishable from the protocol view of I.

If the adversary controls $I = \{A, B_{i_1}, \ldots, B_{i_\ell}\}$, then its view includes aside from its input and output:

- For every B_j such that $j \in \{i_1, \ldots, i_\ell\} \bigcap \{1, \ldots, t\}$, a sequence of encrypted elements: $E_{K_A}(TM(A, B_1) \cdot r_j^1 + q_j^1 \bmod m)$, ..., $E_{K_A}(TM(A, B_{|S|}) \cdot r_j^{|S|} + q_j^{|S|} \bmod m)$.
- $\sum_{i=1}^{|S|} TM(A, B_i) \cdot r_j^i \bmod m$, for every $j = 1, \ldots, t+1$.

$|\{i_1, \ldots, i_\ell\}| \leq t - 1$ because $|I| \leq t$ and therefore there exists some $j \in \{1, \ldots, t\}$ such that $j \notin \{i_1, \ldots, i_\ell\}$. Thus, each set $E_{K_A}(TM(A, B_i) \cdot r_{i_1}^i + q_{i_1}^i \bmod m)$, ..., $E_{K_A}(TM(A, B_i) \cdot r_{i_\ell}^i + q_{i_\ell}^i \bmod m)$, is an encryption of at most $t-1$ random elements in the range $\{0, \ldots, m-1\}$. Since for each $i = 1, \ldots, |S|$, these sets are chosen independently, a simulator can simulate all these encrypted elements by choosing $|S| \cdot |i_1, \ldots, i_\ell|$ random elements in $\{0, \ldots, m-1\}$ and encrypting them.

Similar reasoning works for simulating $\sum_{i=1}^{|S|} TM(A, B_i) \cdot r_j^i + q_j^i \bmod m$, for every $j = 1, \ldots, t$. For an index η such that $\eta \notin \{i_1, \ldots, i_\ell\}$, each sum $\sum_{i=1}^{|S|} TM(A, B_i) \cdot r_j^i + q_j^i \bmod m$ includes a summand q_j^η, which is random (until all the elements $q_1^\eta, \ldots, q_t^\eta$ are summed up). Therefore, the t elements $\sum_{i=1}^{|S|} TM(A, B_i) \cdot r_j^i + q_j^i \bmod m$ can be simulated as t random elements in the range $0, \ldots, m-1$ whose sum modulo m is exactly $\rho(A, x)$. □

5.3 Scheme 3: Improved online communication complexity

A drawback of Algorithm 2 is that if the threshold is linear in the number of participants, $|S|$, then the total communication and the total computational complexity are quadratic in $|S|$. The next scheme improves this result by having two phases, an offline phase and an online phase.

The offline phase occurs only once for each member that enters the system and may be viewed as an initialization phase in which the new member exchanges keys with all other members as a preparation for future information sharing. The online phase occurs whenever a member wishes to compute some other member's reputation. The offline phase requires $O(|S|^2)$ communication and computation, but the online phase requires only linear communication and $O(|S|^2)$ computation.

In this scheme we use a Pseudo-Random Function family, $\mathcal{F} = \{F\}_k$, [12] in addition to homomorphic encryption as in the previous schemes. All the functions in $\mathcal{F}$ have the same domain and range. Additionally, a polynomially bounded adversary can't distinguish between the output of a random function chosen from the family and the output of a completely random function with the same domain and range. There are various theoretical constructions of pseudo-random function families. However, for practical purposes any block cipher can be regarded as such a family, where a function in the family is given by a specific key for the block cipher (such a key determines a function from a block of plaintext to a block of ciphertext).

In the offline phase, each pair of members B_i and B_j chooses a random function $F_{i,j} \in \mathcal{F}$. No one other than B_i and B_j may know the identity of their function. B_i and B_j will each use $F_{i,j}$ in the online phase of the scheme, one of them will use it with a plus sign, and the other with a minus sign. Let $sign_{i,j} = 1$ if $i > j$ and $sign_{i,j} = -1$ if $i < j$.

Algorithm 3 presents the online phase of the scheme. This protocol has two logical stages (although both take place in the same communication round). In the first stage, A and $B_1, \ldots, B_{|S|}$ re-share their values so instead of having to compute a scalar product for $\rho(A, x)$ they have to compute a sum. The second stage involves private and efficient computation of this sum using the functions $F_{i,j}$.

Algorithm 3 Improved communication complexity

1: A runs $G(1^n)$ and obtains a private-public key pair.
2: A chooses a random element RND in the domain of the functions in $\mathcal{F}$.
3: **for all** $B_i \in S$ **do**
4: A sends to B_i the values RND and $E_{K_A}(TM(A, B_i) \bmod m)$.
5: **for all** $B_i \in S$ **do**
6: B_i chooses r_i uniformly at random from $\{0, \ldots, m-1\}$.
7: B_i computes the value $\rho_i \stackrel{\triangle}{=} E_{K_A}(TM(A, B_i) \cdot DTE(B_i, x) - r_i \bmod m)$.
8: B_i computes the value $R_i \stackrel{\triangle}{=} r_i + \sum_{j=1, j\neq i}^{|S|} F_{i,j}(RND) \cdot sign_{i,j} \bmod m$.
9: B_i sends R_i and ρ_i to A.
10: A decrypts the values $\rho_1, \ldots, \rho_{|S|}$.
11: A computes $\rho(A, x) = \sum_{i=1}^{|S|} TM(A, B_i) \cdot DTE(B_i, x) - r_i + R_i \bmod m$.

A computes the correct result because:

$$\sum_{i=1}^{|S|} TM(A, B_i) \cdot DTE(B_i, x) - r_i + R_i \bmod m = \rho(A, x) + \sum_{i=1}^{|S|} \sum_{j=1, j\neq i}^{|S|} F_{i,j}(RND) \cdot sign_{i,j} \bmod m = \rho(A, x)$$

Each value $F_{i,j}(RND)$ appears exactly twice in the sum, once with a plus sign and once with a minus sign, and thus the contribution of each summand $F_{i,j}(RND)$ to the sum is 0.

The communication complexity of steps 3 and 4 is $O(|S| \cdot |E(\cdot)|)$ and the communication complexity of step 9 is $O(|E(\cdot)|)$ for each of $B_1, \ldots, B_{|S|}$. Thus, the total communication complexity of Algorithm 3 is $O(|S| \cdot |E(\cdot)|)$. The total computational complexity is $O(|S|^2)$ times the time required to compute $F_{i,j}(RND)$, due to step 7.

The privacy of the protocol can be shown against any adversary that controls no more than $|S| - 2$ parties. The proof is very similar to the proof of Algorithm 2 with the pseudo-random values $F_{i,j}(RND)$ replacing the truly random values q_j^i in Algorithm 2.

5.4 Faults and a Malicious adversary

The protocols we present assume that the adversary is honest-but-curious. That is, even corrupted parties correctly perform the protocols, but may attempt to discover private information about other members. Our protocols can be adapted to be secure in the presence of a malicious adversary as long as there is an honest majority, by using standard methods, e.g. [13].

Even without malicious intent, users in community-based protocols, such as we propose, may often be absent during protocol execution, thus causing unintentional faults (B_i expects that B_j is part of the computation, but B_j is not present). Our first protocol is robust against such faults while the second and third protocols can be extended to be robust by using secret sharing. We will present these robust protocols in the full version of our paper.

A malicious adversary may do more than simply attempt to subvert the protocols we proposed. An adversary that controls A learns $\rho(A,x) = \sum_{i=1}^{|S|} TM(A,B_i) \cdot DTE(B_i,x)$, where A itself determines the values $TM(A,B_i)$. Setting $TM(A,B_i) = 0$ for any i except for $i = i'$ provides A with the exact value of $DTE(B_i,x)$. If A can repeatedly compute $\rho(A,x)$ then the adversary can learn the private values of many participants.

Even if A proves (by a zero-knowledge protocol) that the values $TM(A,B_i)$ do not form a unit vector, the adversary may still learn all the values $DTE(B_i,x)$. Let M be some non-singular $|S| \times |S|$ matrix. A executes the protocol $|S|$, changing the values of $TM(A,B_1),\ldots,TM(A,B_{|S|})$ to be the value of the next row of M. Multiplying the results by M^{-1} yields $DTE(B_1,x),\ldots,DTE(B_{|S|},x)$.

A possible approach to overcome a malicious adversary is to compute an approximation of $\rho(A,x)$ and achieve differential privacy [6]. The idea is that querying data sets that differ only slightly does not provide information about the data in which they differ. The main tool to achieve differential privacy is by adding noise to the data sets.

In our case, the data sets are the values $TM(A,B_1),\ldots,TM(A,B_{|S|})$ that A holds. The members B_i can add noise to this data in all of our protocols by using the properties of homomorphic encryption.

6 Conclusions

In this paper we discussed the problem of computing reputation of members in a community while preserving the privacy of sensitive information. Such information includes the rating of individual members and the trust that one member has in another. The paper presents three schemes for privacy preserving computation and analyzes their privacy characteristics and communication overhead. The presented schemes apply techniques for secure summation and use these as primitives in a virtual community oriented setting. Our constructions extend state of the art work to elevate existing trust based models in which privacy is a major concern.

Although the reputation computation formula was presented in the context of a specific trust and reputation model, the Knots model, it may be used by

any Trust-set based model which take into account the same sensitive information (members ratings and members trust). In future work we will deal with some other privacy issues not dealt with in the current paper. One issue is the frequency of Update. How one can avoid leakage of private rating information if the reputation is updated frequently (e.g. after every expert/member interaction). Another issue is the private computation and presentation (amount of drill-down) of reputation information in case reputation is accumulated from multiple communities (see [15]).

References

1. ebay, http://www.ebay.com/.
2. J. Audun, I. Roslan, and B. Colin. A survey of trust and reputation systems for online service provision. *Decis. Support Syst.*, 43(2):618–644, March 2007.
3. J. Benaloh. Dense probabilstic encryption. In *Proc. of the Workshop on Selected Areas of Cryptography*, pages 120–128, May 1994.
4. D. Boneh, E. Goh, and K. Nissim. Evaluating 2-dnf formulas on ciphertexts. In *Proc. of TCC*, pages 325–341, 2005.
5. S. Chakraborty and I. Ray. Trustbac: integrating trust relationships into the rbac model for access control in open systems. In *Proc. of the eleventh ACM symposium on Access control models and technologies (SACMAT '06)*, pages 49–58, New York, NY, USA, 2006. ACM.
6. Cynthia Dwork. Differential privacy. In *in ICALP*, pages 1–12. Springer, 2006.
7. N. Gal-Oz, T. Grinshpoun, E. Gudes, and A. Meisels. Cross-community reputation: Policies and alternatives. In *Proc. of International Conference on Web Based Communities (IADIS - WBC2008)*, 2008.
8. N. Gal-Oz, E. Gudes, and D. Hendler. A robust and knot-aware trust-based reputation model. In *Proceedings of the 2nd Joint iTrust and PST Conferences on Privacy, Trust Management and Security (IFIPTM'08)*, volume 263, pages 167–182, Trondheim, Norway, June 2008.
9. C. Gentry. Fully homomorphic encryption using ideal lattices. In *Proc. of STOC*, pages 169–178, 2009.
10. O. Goldreich. *Foundations of Cryptography: Basic Tools*. Cambridge University Press, New York, NY, USA, 2000.
11. O. Goldreich. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, New York, NY, USA, 2004.
12. O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *JACM*, 33(4):792–807, October 1986.
13. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game (extended abstract). In *STOC87*, pages 218–229, 1987.
14. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and systems science*, 28:270–299, 1984.
15. Tal Grinshpoun, Nurit Gal-Oz, Amnon Meisels, and Ehud Gudes. Ccr: A model for sharing reputation knowledge across virtual communities. In *Web Intelligence*, pages 34–41, 2009.
16. E. Gudes, N. Gal-Oz, and A. Grubshtein. Methods for computing trust and reputation while preserving privacy. In *Proc. of DBSEC, Montreal, Canada, 2009. Springer Lecture Notes 5645*, pages 291–298, 2009.

17. J. Yung-Jen Hsu, K. Lin, T. Chang, C. Ho, H. Huang, and W. Jih. Parameter learning of personalized trust models in broker-based distributed trust management. *Information Systems Frontiers*, 8(4):321–333, 2006.
18. A. Josang and R. Ismail. The beta reputation system. In *proceedings of 15th Bled Electronic Commerce Conference e-Reality: Constructing the e-Economy*, June 2002.
19. L. Mui, M. Mohtashemi, and A. Halberstadt. A computational model of trust and reputation for e-businesses. In *Proc. of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 7*, page 188, Washington, DC, USA, 2002.
20. D. Naccache and J. Stern. A new public key cryptosystem based on higher residues. In *Proc. of ACM Conference on Computer and Communications Security*, pages 59–66, 1998.
21. J. Nin, B. Carminati, E. Ferrari, and V. Torra. Computing reputation for collaborative private networks. In *33rd IEEE Int. COMPSAC conference*, pages 246–253, 2009.
22. T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. In *Proc. of EUROCRYPT*, pages 308–318, 1998.
23. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proc. of EUROCRYPT*, pages 223–238, 1999.
24. E. Pavlov, J. S. Rosenschein, and Z. Topol. Supporting privacy in decentralized additive reputation systems. In *Proc. of 2nd Intl. Conf. on Trust Mgmt (iTrust'04)*, pages 108–119, 2004.
25. T.P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Proc. of CRYPTO*, pages 129–140, 1991.
26. F. Pingel and S. Steinbrecher. Multilateral secure cross-community reputation systems for internet communities. In *proceedings of TrustBus (TrustBus '08)*, pages 69–78, 2008.
27. P. Resnick and R. Zeckhauser. Trust among strangers in Internet transactions: Empirical analysis of eBay's reputation system. In Michael R. Baye, editor, *The Economics of the Internet and E-Commerce*, volume 11 of *Advances in Applied Microeconomics*, pages 127–157. Elsevier Science, 2002.
28. J. Sabater and C. Sierra. Review on computational trust and reputation models. *Artificial Intelligence Review*, 24(1):33–60, 2005.
29. A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.
30. S. Steinbrecher. Design options for privacy-respecting reputation systems within centralised internet communities. In *Proc. of the IFIP TC-11 21st International Information Security Conference (SEC 2006)*, pages 123–134, 2006.
31. Zenggang Xiong, Yang Yang, Xuemin Zhang, Dairong Yu, and Li Liu. A trust-based reputation system in peer-to-peer grid. In *HCI (15)*, pages 228–235, 2007.
32. B. Yu and M. Singh. Detecting deception in reputation management. In *proceedings of Second International Joint Conference on Autonomous Agents and Multi-Agent Systems*, pages 73–80, 2003.
33. Bin Yu, Munindar P. Singh, and Katia Sycara. Developing trust in large-scale peer-to-peer systems. In *Proceedings of First IEEE Symposium on Multi-Agent Security and Survivability*, pages 1–10, 2004.