

A Verifiable, Centralized, Coercion-Free Reputation System

Florian Kerschbaum
SAP Research
Karlsruhe, Germany
florian.kerschbaum@sap.com

ABSTRACT

Reputation systems are popular tools to evaluate the trustworthiness of an unknown party before a transaction, but the reputation score can greatly impact the rated subject, such that it might be inclined to suppress negative ratings. In order to elicit coercion-resistant, honest feedback, this paper proposes a reputation system that provides complete privacy of the ratings, i.e. neither the ratee nor the reputation system will learn the value of the rating. We take both, a cryptographic as well as a non-cryptographic approach, to the problem. Privacy of ratings may foster bad mouthing attacks where an attacker leaves intentionally bad feedback. We limit the possibility for this attack by providing a token system such that one can only leave feedback after a transaction, and provide a cryptographic proof of the privacy of our system. We consider the Virtual Organization formation problem and develop and evaluate a novel reputation aggregation algorithm for it.

Categories and Subject Descriptors

H.5.3 [Information Interfaces and Presentation]: Group and Organization Interfaces—*Collaborative computing*; D.4.6 [Operating Systems]: Security and Protection—*Cryptographic controls*

General Terms

Algorithms, Security

Keywords

Reputation Systems, Privacy, Rater Privacy

1. INTRODUCTION

Reputation systems have often been proposed to support Virtual Organizations (VO) formation [3, 18, 28, 34]. A common problem that arises in these and other reputation systems is that the evaluated parties exert pressure to suppress negative ratings. One means to blackmail another

party is to threaten with negative ratings in return for negative ratings. It does not even have to be case that the reputation system accepts only mutual ratings, but as in the case of VO formation [34] where all parties are raters and ratees, one can retaliate in a seemingly unrelated transaction. The solution presented in this paper is to protect the privacy of the ratings, i.e. ratings are never revealed to anybody. If a rated party never learns how another party rated it, it cannot exert pressure on that particular party.

Many design choices and considerations need to be taken into account when designing a reputation system that protects the privacy of the ratings. We briefly review the main choices made in this paper and discuss their impact. In particular, we aim to remove the possibility of any party to breach the privacy of the ratings and furthermore protect the system against negative implications of false private ratings.

If the ratings remain private this opens the system to “bad-mouthing” attacks where parties leave unsubstantiated bad feedback in order to deliberately lower the reputation of a particular party. We prevent these attacks by only allowing feedback by parties which have received a cryptographic token by the rated party that they have engaged with in a transaction. Such a token is bound to one’s identity and non-transferable, such that only legitimate “customers” can leave feedback. The only way an adversary can launch a bad-mouthing attack is to make the victim engage in a transaction with everyone in its coalition. In particular we cannot rely on anonymization techniques, such as anonymous communication or credentials, for the transaction, since the VOs are used for real world business transactions where laws and the flow of physical goods require identity disclosure.

A further problem with private ratings is that although the rating itself may be protected by cryptographic protocols, its impact is almost always public. A positive rating raises the reputation score and a negative rating lowers the reputation score. The reputation score is public, since it must be accessible at any time a transaction can take place. The rated party knows the point in time when a transaction ended and the other party can leave feedback for him. It retrieves its reputation score before and after and infers from the change the left feedback.

The solution to this problem is obvious: multiple ratings have to be combined before recomputing the reputation score. An aggregate score may only be published after a threshold of new individual ratings has been exceeded. We empirically evaluate this design option.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

WPES’09, November 9, 2009, Chicago, Illinois, USA.

Copyright 2009 ACM 978-1-60558-783-7/09/11 ...\$10.00.

In a centralized reputation system implemented in only one organization the reputation service provider is able to breach all privacy mechanisms. He is able to compute or retrieve the reputation at any time, such that e.g. the combination of ratings does not apply to him. We split the central reputation system into two systems that are mutually distrustful. One is responsible for the publishing of the reputation scores and the other for the collection of feedback. In our use case of VO formation one can be the reputation service provider and the other a consumer privacy organization.

1.1 Problem Background

The problem of selecting a partner for VO formation [34] is a hard classification problem. Many companies provide their services often offering virtually equivalent service levels. Companies that have low performance go out of business quickly, since customers do not return and the set of potential customers is limited. The remaining companies have very similar performance levels, such that the task of a reputation system is to separate the excellent ones from the very good ones.

In the light of this tough competition it is understandable that companies push their customers to leave the best feedback possible and try to suppress negative ratings with almost all measures possible. Our approach is therefore to entirely protect the rating of the customer and thereby elicit an honest, coercion-resistant rating.

Compared to related work we contribute a reputation system that

- is centralized enabling the corresponding business model of providing a reputation service.
- is the first to keep the value of ratings private from ratee and reputation system.
- prevents anyone from leaving feedback without a transaction and thereby prevents bad mouthing attacks.
- does not require a central registry of transactions enabling it to be used in an open community.
- does not require transactions to be anonymous enabling it to be used for business transactions.

The remainder of the paper is structured as follows: First we review related work in the next section. Then we introduce some of the cryptographic techniques used by our system in Section 3, before we describe our system architecture and security mechanism in Section 4. In this section we also give a proof of privacy of our mechanism reducing it to a well-known hard problem. Different feedback aggregation algorithms are described and evaluated in Section 5. In Section 6 we analyze the parameters to keep the rating private, even if the reputation score of the party is public. Section 7 presents our conclusions.

2. RELATED WORK

The problem of privacy has been widely studied for reputation systems. In [35] a good introduction to the privacy problem in reputation systems is given. The main problem described is to control the user's private data, i.e. its identity and ratings. Their use and collection under European

legislation need to be determined by the user. This is confirmed in a legal study of privacy and reputation systems in [22]. The proposed solution is to use pseudonyms, but then one has to handle resulting attacks such as white-washing (creating new identities).

In [37] a reputation system for mobile data collection and exchange is presented. The privacy protection extends to protecting the identity and rating from the rated party using a trusted device and a trusted third party. Obviously a trusted third party can solve all privacy problems, but finding such a party is difficult and raters might be reluctant to leave honest, negative feedback, if even one party learns that it leaves such feedback. In our solution there is no single trusted third party. Instead we distribute the reputation system functionality over two parties that are mutually distrustful.

In [19] security and privacy protection using trusted computing is presented. The privacy protection extends to the protection of identity of the raters, i.e. ratings are not protected. It is well known that a secure hardware solution can implement a trusted party, such that the results are interesting, but not surprising. We protect the rating itself and do not use a trusted third parties.

In [2] anonymity of ratee and rater even towards the reputation system is achieved by using anonymous credentials [9, 10, 21]. This does not protect the privacy of the rating and the system only allows for positive ratings anyway, such that the value of any ratings is known. Negative ratings could be suppressed, since the ratee has to submit the feedback. In [24] a system for anonymous peer review is presented where reputation is aggregated based on accepted papers and later used to weigh review results. While the concept is very interesting, it also only allows for positive ratings in its reputation system.

In [32] a reputation system for multiple communities is presented. It uses a hierarchy of pseudonym systems, such that the reputation is bounded to a higher level pseudonym, but not the identity. This still does not protect the privacy of the rating which can be read by the reputation system and inferred by the ratee and therefore does not achieve our goal of coercion-resistance. Also the system relies on the score to be public in order to have some verifiability of the reputation score computation. In [6] reputation is also left anonymously. The pseudonym for the reputation system is protected in a smart-card. It suffers from the same problems as [32]. We prevent both attacks in our reputation system.

In [12] a content recommendation system based on privacy-preserving data mining [1, 20] techniques is presented. In a recommendation system the problem of protecting the ratings does not so much arise when aggregating the data, but rather when retrieving the recommendation, since the recommendation depends on personal preferences. Therefore the main focus of the paper is to present a private retrieval protocol that allows to adapt to the personal preferences without revealing them. In our case the focus is rather on secure aggregation protecting the ratings, such that honest ratings are possible.

The aggregation of data is protected as well in [31], but they only allow distributed reputation systems, such that traditional secure multi-party computation [5, 14, 38] techniques are applicable. They present a number of secure addition protocols that protect the individual ratings. Our reputation system extends that to a centralized reputation

systems and combines it with a novel mechanism to prevent bad mouthing. Furthermore we analyze the dependence of the final reputation on the individual ratings, such that if the reputation is public, we can still have private ratings. A more sophisticated distributed system for private collaborative filtering for recommendation is given in [11].

The problem of Virtual Organization formation has been already considered in [3, 18, 28]. Reputation systems are considered as a useful tool to improve selection performance.

Different requirements for these reputation systems are presented. In [18] security of the reputation system is considered. It presents a new reputation system more secure against collusion. Similar results have been achieved with a different mechanism in [36]. A general overview of possible attacks on reputation systems can be found in [33].

Our cryptographic mechanisms are based on bilinear maps and the security assumptions made popular by [17]. Our token scheme is very reminiscent of the short signatures based on bilinear maps in [8].

A commonly applied technique for privacy in reputation systems is to base the reputation system on anonymity techniques, in particular anonymous credentials [2, 6, 24, 32]. Anonymous credentials rely on the principles of pseudonymization. A transaction and a rating are left under a pseudonym and for a pseudonym linkable only by certain parties. A straight-forward application of the primitives of anonymous credentials [9, 10, 21] is not able to achieve the linkability required by our system architecture (see Section 4) designed to achieve privacy of the contents of the rating. Nevertheless a modification of the principles of anonymous credentials could serve our purposes and we are introducing our specialized token system based on the Bilinear Decisional Diffie-Hellman Assumption.

We also identified practical, legal and technical problems in applying pseudonyms in our application scenario. First, in order to achieve full unlinkability between rater and rating by any party as in our system, pseudonyms would need to be renewed for every transaction. Besides the necessary overhead, this contradicts the binding of reputation to identity and fosters the white-washing attack of inventing new identities. This problem has been recognized and the reputation systems in the literature try to circumvent it; either by limiting the pseudonym renewal [6, 32] or by limiting the feedback to only positive feedback [2, 24]. When there is no negative feedback, there is no incentive to renew one's identity, but negative feedback is crucial for comparing business with different types of transaction volumes as in a VO breeding environment. When a pseudonym cannot be renewed, transaction and feedback often become linkable by re-identification attacks, as seen in many recent examples [25, 26]. Second, we intend to apply our reputation system to VOs for business transactions. In a business transaction, even for electronic goods or services, pseudonymization is often unacceptable due to the governing law. In transactions involving physical goods the real identity often cannot be hidden. Therefore pseudonyms and anonymous communication are not applicable to our application scenario.

In an admittedly simplifying summary, the approach of privacy-preserving reputation systems in the literature [2, 6, 19, 24, 32, 37] is to protect *who* rated you, instead we protect *how* he rated you.

3. PRELIMINARIES

3.1 Homomorphic Encryption

Our reputation system is built using homomorphic encryption. In homomorphic encryption one operation on ciphertexts produces an encryption of the result of a homomorphic operation on corresponding plaintexts. In particular, we require the homomorphic operation to be addition (modulo a key-dependent constant). Several such encryption systems exist [4, 13, 23, 29, 30]. We recommend Paillier's encryption system [30] for implementation. Let $E_X(x)$ denote the encryption of x with X 's public key and $D_X()$ the corresponding decryption with X 's private key, then Paillier's encryption system has the following property:

$$D_X(E_X(x) \cdot E_X(y)) = x + y$$

With simple arithmetic the following property can be derived

$$D_X(E_X(x)^y) = x \cdot y$$

3.2 Cryptographic Pairings

Given a security parameter k , let \mathbb{G} and \mathbb{G}' be two groups of order p for some large prime p , where the bit-size of p is determined by the security parameter k . Our scheme uses a computable, non-degenerate bilinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}'$ for which the *Bilinear Decisional Diffie-Hellman Problem (BDDH)* problem is assumed to be hard. In what follows, we denote $\mathbb{Z}_p^* = \{1, \dots, p-1\}$.

Modified Weil or Tate pairings on supersingular elliptic curves are examples of such maps. Recall that a bilinear pairing satisfies the following three properties:

- Bilinear: for $g, h \in \mathbb{G}$ and for $a, b \in \mathbb{Z}_p^*$, $\hat{e}(g^a, h^b) = \hat{e}(g, h)^{ab}$
- Non-degenerate: $\hat{e}(g, g) \neq 1$ is a generator of \mathbb{G}'
- Computable: there exists an efficient algorithm to compute $\hat{e}(g, h)$ for all $g, h \in \mathbb{G}$

4. SYSTEM ARCHITECTURE

In this section we describe the system architecture without reference to any reputation score aggregation algorithm. In the next section we will describe how to use the system for two specific aggregation algorithms.

Let SP_1 be the first reputation service provider and SP_2 be the second reputation service provider. We denote \mathbb{X} the set of all ratees and raters, i.e. all parties can rate as well as be rated. For simplicity we assume binary ratings $z \in \{0, 1\}$ in this section where 1 denotes a positive rating and 0 a negative rating. Our system can be extended to ratings within a finite range, but then the notion of a negative rating may become more unclear. Fundamentally a feedback r is a triple $r = \langle X_{ratee} \in \mathbb{X}, X_{rater} \in \mathbb{X}, z \in \{0, 1\} \rangle$. \mathbb{R} is the multiset of all left feedback r_i . A reputation score for X is a function $f(\mathbb{S}_X)$ of the ratings of the selection $\mathbb{S}_X = \{r_i | r_i \in \mathbb{R} \wedge r_i.ratee = X\}$.

An overview of our reputation system is depicted in Figure 1 and its steps proceed as follows.

1. Alice (A) and Bob (B) engage in a transaction. As a result of that transaction Alice issues Bob a token, that

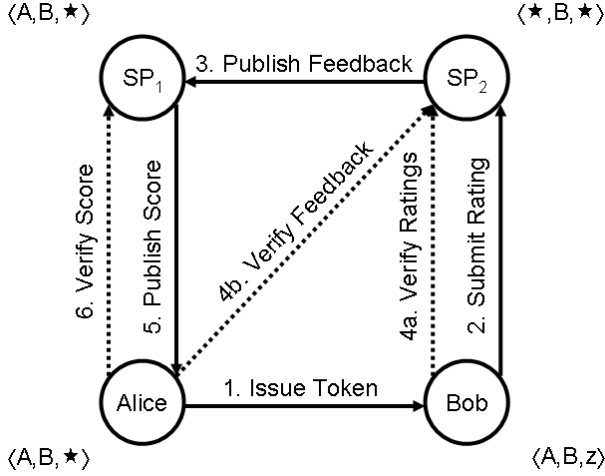


Figure 1: Architecture

he can use to leave feedback. This token enables privately correlating feedback, transactions and involved parties by the appropriate actors. We emphasize that Alice should issue the token before the result of the transaction is known. Otherwise she could refuse to issue one, if the result was negative and prevent Bob from leaving negative feedback. Bob should not engage in any transaction without having received the token for feedback first. Since Alice tries to attract customers, this incentivizes her to issue tokens.

2. Bob leaves his feedback rating with SP_2 .
3. SP_2 collects feedback from several raters and eventually publishes all feedback on a public bulletin board. The collection of feedback is necessary in order to prevent the ratee from inferring individual ratings from the impact on its reputation score. We detail the parameters of that collection later in Section 6.
4. All ratees verify in the published feedback that no feedback for them is present for which they did not issue a token. All raters verify in the published feedback that each rating is as they left it.
5. SP_1 computes the aggregate reputation score for each ratee and publishes it in the same bulletin board.
6. All ratees verify that SP_1 has computed their score as prescribed by the algorithm and according to the left feedback.

4.1 Security Requirements

GOAL 1. No party $X \in \mathbb{X} \cup \{SP_1, SP_2\}$ should learn any feedback $r_i \in \mathbb{R} | r_i.X_{\text{rater}} \neq X$ that was left by another party either by disclosure or by inference.

GOAL 2. No party $X \in \mathbb{X} \cup \{SP_1, SP_2\}$ should be able to inject any negative feedback $r_i \in \mathbb{R} | r_i.X_{\text{rating}} = 0$ not related to any transaction without being detected.

Our main security assumption is that no one colludes with any of the service providers SP_1 and SP_2 , including themselves. Both our security goals can be broken in our system in case of such a collusion.

We emphasize that we do not exclude collusion between ratees and raters. Such collusion is very difficult to prevent or even detect, but usually it can only be formed to forge positive feedback, not negative feedback. We prevent any coalition from leaving feedback outside the colluders' group, but cannot prevent them from leaving feedback inside the group. Nevertheless leaving negative feedback inside the group is not supported by any incentives, since only the outsiders of the group benefit.

We do not rely on a central registry for transactions. Only the parties involved in a transaction need to know about it. It then becomes difficult, if not even impossible, to restrict collusion for forged positive feedback. The consequences of forged positive feedback can be handled by the reputation aggregation algorithm, e.g. [18].

We will explain how we achieve our security goal 1 by describing the involved parties' view of the feedback. Alice (A) and Bob (B) have engaged in a transaction, i.e. our feedback r is $r = \langle A, B, z \rangle$. We will denote a value unknown to a party by \star . Alice knows about the transaction and therefore inevitably has the view $r = \langle A, B, \star \rangle$, i.e. we need to prevent Alice from learning z . This is also the main motivation of our paper, since Alice learning the rating z implies her ability to coerce Bob to leave positive feedback. We also prevent Alice from learning z by inference about changes of her reputation score.

Bob creates feedback and obviously has the full view $r = \langle A, B, z \rangle$. The second service provider SP_2 collects feedback from the raters, i.e. it interacts with Bob. It has the view $r = \langle \star, B, \star \rangle$. The first service provider SP_1 aggregates feedback by computing the function $f(\mathbb{S}_A)$. It has the view $r = \langle A, \star, z \rangle$. Charlie (C), as an outsider to the transaction, remains oblivious having view $r = \langle \star, \star, \star \rangle$.

We argue that these are the minimal views necessary for the parties to perform their functions. SP_2 needs to be able to replace Bob's feedback in our aggregation algorithm and SP_1 needs to select the feedback for a ratee in the reputation score computation. Therefore SP_2 needs to be able to link feedback and rater and SP_1 needs to be able to link feedback and ratee.

We break our security goal 2 into six subgoals.

- **Unforgability by Rater:** No party $X \in \mathbb{X}$ should be able to forge a token related to feedback for another party $X' \in \mathbb{X}$.
- **Unforgability by Service Provider:** The service provider SP_2 should not be able to forge a feedback submission by any party X .
- **Verifiability by Ratee (1):** A ratee X should be able to identify all published feedback linked to his identity and verify that they are related to a recorded transaction.
- **Verifiability by Ratee (2):** A ratee X should be able to identify all published feedback linked to his identity and verify that they are linked to the transaction partner X' .
- **Verifiability by Ratee (3):** A ratee X should be able to identify all published feedback linked to his

identity and verify that service provider SP_1 has computed its reputation score according to them.

- **Verifiability by Rater:** A rater X should be able to identify all published feedback linked to his identity and verify that the rating is as he left it.

4.2 Protocols

We assume the existence of a unique identity, e.g. through a public key infrastructure (PKI). We denote by $S_X()$ a signature using the private key of party X . Furthermore we assume that all communication is done over secure and authenticated channels which can e.g. be established using the PKI.

The public parameters of our system are groups \mathbb{G} and \mathbb{G}' of order p , a generator g of \mathbb{G} and a bilinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}'$. The service providers publish these parameters before establishing their service. Furthermore the first service provider SP_1 chooses a private, public key pair in a homomorphic encryption scheme as described in Section 3.1 and publishes its public key $E_{SP_1}()$.

4.2.1 Registration

When Alice wants to register with the reputation service providers, she randomly chooses two secret keys $s_A \in \mathbb{Z}_p$ and $t_A \in \mathbb{Z}_p$. She sends her public keys g^{s_A} to SP_1 and g^{t_A} to SP_2 . SP_1 publishes a list with all public keys g^{s^X} and their identities X or alternatively issues a certificate to Alice. In the same manner SP_2 does for g^{t^A} .

4.2.2 Token Issue

Alice and Bob intend to engage in a transaction and Alice issues Bob a token for him later leaving feedback at the reputation service. Alice chooses a random number $r \in \mathbb{Z}_p$ and sends to Bob $\alpha = g^r$, $\beta = g^{r^{s_A}}$, $\gamma = g^{r^{t_B}}$, $S_A(g^r, g^{r^{s_A}}, g^{r^{t_B}})$. Bob verifies that $\hat{e}(\alpha, g^{s_A}) = \hat{e}(\beta, g)$ and that $\hat{e}(\alpha, g^{t_B}) = \hat{e}(\gamma, g)$. Alice keeps a copy of r along with the record of the transaction.

Note that we realize that except for the signature Bob can create re-randomized replicas of the token or forge the entire token by himself. Yet the signature reveals Alice's identity and we therefore cannot use it in the communication with SP_2 . Instead we exploit the re-randomization feature to make the token unlinkable for SP_2 and rely on Alice identifying any feedback forged by Bob.

4.2.3 Feedback Submission

Bob chooses his rating z and encrypts it in the homomorphic encryption scheme $E_{SP_1}(z)$. He also uniformly chooses two random numbers $l \in \mathbb{Z}_p$ and $m \in \mathbb{Z}_p$. He sends $\delta = g^r, g^l, E_{SP_1}(g^{r^l}), g^{r^{s_A l}}, \epsilon = g^m, \zeta = g^{r^m}, \eta = g^{r^{t_B m}}, \theta = E_{SP_1}(z), S_B(g^r, g^l, E_{SP_1}(g^{r^l}), g^{r^{s_A l}}, g^m, g^{r^m}, g^{r^{t_B m}}, E_{SP_1}(z))$ to the second service provider SP_2 . SP_2 verifies that $\hat{e}(\delta, \epsilon) = \hat{e}(\zeta, g)$ and that $\hat{e}(\zeta, g^{t_B}) = \hat{e}(\eta, g)$.

4.2.4 Feedback Publication

The second service provider SP_2 publishes all collected feedback in intervals. We justify the delay and determine the length of the interval in Section 6. For each feedback SP_2 publishes $\iota = g^r, \kappa = g^l, \lambda = E_{SP_1}(g^{r^l}), \mu = g^{r^{s_A l}}, \nu = g^m, \xi = g^{r^{t_B m}}, o = E_{SP_1}(z)$.

Alice scans all feedback and checks whether $\hat{e}(\iota, \kappa)^{s_A} = \hat{e}(\mu, g)$. If it is true, Alice concludes that this feedback is

left for him and will be used in her reputation score computation. She also checks whether she has a record r matching $g^r = \iota$. She then recalls that Bob was the rater of the transaction related to r and verifies that $\hat{e}(\nu, g^{t_B})^r = \hat{e}(\xi, g)$. If any check fails, she claims that the feedback is forged and initiates an investigation as we describe it in Section 4.2.6.

Bob could similarly scan all feedback and check whether $\hat{e}(\iota, \nu)^{t_B} = \hat{e}(\xi, g)$, but he performs an inverse check. He verifies that all his submitted feedback are present by comparing their identifier $\delta = \iota$ and checks whether all values match the ones he submitted, in particular whether his rating is unchanged $o = \theta = E_{SP_1}(z)$. If any check fails, he similarly claims a forged feedback.

4.2.5 Reputation Score Computation

The first service provider SP_1 reads all published feedback $\iota = g^r, \kappa = g^l, \lambda = E_{SP_1}(g^{r^l}), \mu = g^{r^{s_A l}}, \nu = g^m, \xi = g^{r^{t_B m}}, o = E_{SP_1}(z)$. SP_1 decrypts $D_{SP_1}(\lambda) = \pi = g^{r^l}$ and checks whether $\hat{e}(\iota, \kappa) = \hat{e}(\pi, g)$. Then SP_1 iterates through the list of g^{s^X} and checks whether $\hat{e}(\pi, g^{s^X}) = \hat{e}(\mu, g)$. If it is true, SP_1 concludes that $X = A$, i.e. it should use this feedback in the computation of the reputation score of Alice. SP_1 decrypts $z = D_{SP_1}(o)$, uses z to compute the reputation score and publishes that score along with Alice's identity. SP_1 claims a forged feedback, if it does not find any corresponding g^{s^X} and cannot use it in any score computation.

The service provider SP_1 must produce and publish a zero knowledge proof (ZKP) for the correct computation of the score from the ciphertexts o . We show in Section 5.1 an efficient proof for our aggregation algorithms. Alice who knows all feedback left for her from the previous step in Section 4.2.4 verifies the ZKP by SP_1 .

4.2.6 Dispute Resolution

A process of dispute resolution needs to be established in order to identify the originator of a published, forged feedback. This is particularly important, since both the service provider SP_2 and Bob can forge arbitrary published feedback. Detection is our mechanism for deterring such behavior. We also take into account malicious behavior by Alice, e.g. by issuing a token for leaving feedback for another party; although this can be detected by Bob before the feedback has been published.

A variation of the forgery attack that may be only of importance for some aggregation algorithm is the transfer of a token. In the aggregation algorithm presented in Section 5.3, the number of ratings by one party is limited. A party might therefore be inclined to forward a token to another party which then leaves feedback on its behalf. Note that in this case the identifier g^r of the feedback is valid. We counter this attack by the rater's verifiability of the source of feedback left for him. The same dispute resolution process will be initiated and both parties can be detected.

If any party, be it the service provider SP_1 , Alice or Bob claim that any feedback has been forged, a trusted third party D is called upon. Each party presents as evidence the published feedback and the judge D decides which party is at fault. For each claim, the order of accused is first SP_2 , second B , and finally A . If a party can prove its innocence the next party will be accused, i.e. if SP_1 and Bob can prove their innocence Alice will be convicted.

The service provider SP_2 's proof is the signature $S_B(g^r, g^l, E_{SP_1}(g^{r^l}), g^{r^{s_A l}}, g^m, g^{r^m}, g^{r^{t_B m}}, E_{SP_1}(z))$ submitted by Bob.

D verifies the equality of each entry in the signature with the published feedback and if all checks succeed it accepts the proof. Bob's proof is the signature $S_A(g^r, g^{r^{s_A}}, g^{r^{t_B}})$ received with the token. D verifies that $g^r = \iota$, $\hat{e}(\kappa, g^{r^{s_A}}) = \hat{e}(\mu, g)$ and that $\hat{e}(\nu, g^{r^{t_B}}) = \hat{e}(\xi, g)$. If all checks succeed, it accepts the proof.

We emphasize that this implies that Alice will be detected, if she falsely claims a forged feedback in order to erase a negative rating.

4.2.7 Leaving Self Feedback

While it is not our goal to prevent collusion from generating forged positive feedback, it might be necessary to prevent a party from leaving positive feedback for itself in many aggregation algorithms, since there is no transaction cost. The confidentiality in the parties' views implies that no party alone can decide whether Bob has left feedback for himself.

The straight-forward solution would be to increase the service provider SP_2 's view to include the ratee, e.g. by sending g^{r^l} instead of its ciphertext $E_{SP_1}(g^{r^l})$, but there exists a more privacy-preserving solution. Unfortunately it significantly complicates the description of our algorithm, such that we will only briefly overview the idea.

The basic idea is to have Bob publish $g^{s_B t_B}$ which can be verified by checking $\hat{e}(g^{s_B}, g^{t_B}) = \hat{e}(g^{s_B t_B}, g)$. Bob submits another value $g^{r^{2lm}}$ with his feedback. SP_2 then checks if $\hat{e}(g^{r^{ls_A}}, g^{r^{mt_B}})$ and $\hat{e}(g^{r^{2lm}}, g^{s_B t_B})$ differ. The remaining problem is that SP_2 cannot verify if $g^{r^{2lm}}$ has been computed correctly, since it does not know g^{r^l} , and Bob could therefore simply submit a random value. It therefore needs to enlist the help of SP_1 , but without SP_1 being able to link the feedback to g^{t_B} either, which revealing g^{r^m} would do. SP_2 does so by choosing a random number $n \in \mathbb{Z}_p$ and publishing $g^{r^{mn}}$ and $g^{r^{2lmn}}$ along with the feedback. Note that SP_2 must even keep g^n secret. SP_1 then can check whether $\hat{e}(g^{r^l}, g^{r^{mn}}) = \hat{e}(g^{r^{2lmn}}, g)$.

The dispute resolution process ensures that SP_2 cannot frame Bob for submitting feedback for himself.

4.3 Security

4.3.1 Bilinear Decisional Diffie-Hellman Assumption

For any probabilistic polynomial-time algorithm \mathcal{A} , we define \mathcal{A} 's advantage in solving the Bilinear Decisional Diffie-Hellman (BDDH) problem in \mathbb{G} with group generator g as

$$Adv_{\mathcal{A}}(k) = |\Pr[\mathcal{A}(\mathbb{G}, g, g^a, g^b, g^c, g^x) = \top \text{ if } x = abc] - \frac{1}{2}|$$

DEFINITION 1. We say that the BDDH assumption holds for \mathbb{G} , if for any probabilistic polynomial-time algorithm \mathcal{A} , $Adv_{\mathcal{A}}(k)$ is a negligible function of k .

The BDDH assumption is a standard assumption in cryptographic literature, e.g. [7, 17].

4.3.2 Confidentiality

The rating z remains confidential between the encipherer Bob and the holder of the private key SP_1 . For a proof of ciphertext security we refer the reader to the constructions of the encryption schemes [4, 13, 23, 29, 30]. We recommend a cipher that is semantically secure against chosen plaintext attacks (IND-CPA).

4.3.3 Unlinkability

There are certain unlinkability requirements defined in the parties' views, e.g. SP_1 may not be able to link a feedback to the public information g^{t_B} of its rater Bob. All our unlinkability properties rely on the BDDH assumption and the same mechanism achieves these properties.

If we abstract from the individual values a party X is given a triple g^a, g^b, g^c and needs to decide whether g^x is g^{abc} . An example is the second service provider SP_2 is sent $g^a = g^r, g^b = g^l, g^c = g^{r^{ls_A}}$ by Bob. SP_2 can retrieve $g^c = g^{s_A}$ for Alice from SP_1 , i.e. if SP_2 can decide whether $x = abc$ it can link the submitted feedback to Alice. It is easy to define a game in which an adversary given such information can be used to solve a BDDH challenge if he successfully links the feedback. Thereby one can prove unlinkability by reduction to BDDH. We omit the details for brevity.

The unlinkable parties are the rater Bob for SP_1 , the ratee Alice for SP_2 , and both rater Bob and ratee Alice for outsider Charlie. In all cases the setup is similar and almost identical games can be defined. This completes the proof sketch of unlinkability in the parties' views as defined in Section 4.1.

We make some feedback linkable for certain parties, e.g. SP_2 can link feedback to raters, by revealing an intermediate value, e.g. $g^{ab} = g^{r^m}$. The party can then use the bilinear map to solve the BDDH problem.

4.3.4 Unforgability

We have defined two unforgability requirements in Section 4.1: one for the rater and one for the service provider SP_2 . Both are realized using standard public-key signatures. We refer the reader to the literature, e.g. [8], for an appropriate signature scheme. Note that the signatures are not published or forwarded which would reveal the identity of the sender and therefore no anonymity requirements need to be met by the signature scheme.

4.3.5 Verifiability

Our verifiability requirements require the verifiers to link certain values by computing the bilinear map. The underlying assumption is that random choices, such as identifiers g^r , randomization values g^l, g^m and keys g^{s_A}, g^{t_B} are unique. A collision of two random choices in a set of size n follows the birthday attack and occurs with probability roughly $\frac{1}{2}$ if $n = \sqrt{p}$. This is still a negligible function of the security parameter k .

A second verification is performed using a ZKP. We give the details of this check in the next section, since it depends on the aggregation algorithm.

5. AGGREGATION ALGORITHMS

5.1 Zero Knowledge Proof

In [13] an efficient ZKP for the encryption schemes [13, 30] has been described. Given a ciphertext $E_{prover}(x)$ and its plaintext x a prover can prove in zero knowledge (i.e. without revealing the private key) that x is the correct plaintext for $E_{prover}(x)$. Loosely speaking, the prover computes the randomness u used to generate the ciphertext. The verifier encrypts x using u resulting in $E'_{prover}(x)$ and compares $E_{prover}(x)$ and $E'_{prover}(x)$ for equality.

5.2 Beta Reputation System

The first aggregation algorithm we consider is the Beta reputation system [16] in its simplest form with binary positive or negative values and no discounting or forgetting. Let r be the number of positive ratings and s be the number of negative ratings. Then the reputation score can be written as $\frac{r+1}{r+s+2}$.

As described in Section 4.2.3 Bob encrypts $z = 1$ for a positive rating and $z = 0$ for a negative rating in the homomorphic encryption system. The second service provider should request a ZKP from Bob that the ciphertext is indeed an encryption of one of the two values. Efficient methods for doing so exist, see [13].

We used a homomorphic encryption system for the rating, since it allows an efficient ZKP as described in Section 5.1. Let $\mathbb{Z} = \{z_1, \dots, z_{|\mathbb{Z}|}\}$ be the set of ratings left for Alice. The service provider SP_1 computes a ciphertext of the number r of positive ratings as $E_{SP_1}(r) = \prod_{i=1}^{|\mathbb{Z}|} E_{SP_1}(z_i) = E_{SP_1}(\sum_{i=1}^{|\mathbb{Z}|} z_i)$. SP_1 decrypts $E(r)$ generating the ZKP u . It publishes Alice's identity, the score $t = \frac{r+1}{|\mathbb{Z}|+2}$ and u .

Alice can identify all ciphertexts in \mathbb{Z} . She also computes the ciphertext $E_{SP_1}(r)$, the number of ratings $|\mathbb{Z}|$ and the number of positive ratings $r = \frac{t}{|\mathbb{Z}|+2} - 1$. She can then verify the ZKP u .

5.3 Yet Another Reputation System

For our problem, VO formation, we need to design a reputation system with special properties. Our concerns are to level the impact individuals can have and to provide the dynamic of ratings required.

Some VOs are formed quickly on demand and short-lived, e.g. a learning service provider for a composite training course, while others are long-lived and stable, e.g. an airplane engineering VO. Therefore they vary significantly in number and also transaction volume, neither of which is necessarily indicative of relevance. As a consequence we chose to democratize the ratings, such that a party that engages in many transactions has the same impact as a party with few transactions and the ratings of each party are equally important.

Our reputation system is a combination of a local aggregation algorithm which is freely chosen by the rater and a global aggregation algorithm. The local aggregation algorithm outputs a fixed number of ratees and ratings, no matter how many or what kind of transactions a party has performed. The global aggregation system combines all these local ratings.

The local aggregation system allows the rater to choose any strategy for submitting its ratings. An example would be a distinction between a failed project due to negligence or simply unfriendly staff. Also the local aggregation allows a party to quickly adapt its ratings and therefore provides the necessary dynamic that global aggregation algorithms often lack.

5.3.1 Algorithm

The parties perform setup, registration and transaction with token issue as described in Section 4.2. After locally aggregating his feedback Bob may at his discretion leave n positive ratings and m negative ones where n and m are system-wide parameters. Bob may also choose not to leave feedback for any of those $n + m$ ratings, i.e. they maybe

empty, but he has to break ties and make a decision on the ranking.

We chose the following weights for ratings: Positive ratings linearly decrease in their value from n to 1 in steps of 1. Negative ratings exponentially decrease from -1 to $-(4^{m-1})$ in steps of multiples of 4. The reason to weigh negative ratings higher (and with a steeper decrease) is to increase the ability to distinguish between similarly performing parties that have few negative ratings.

Bob has to store and send the same re-randomized tokens as in Section 4.2.3 for each of the feedback. Bob also sends one of the ciphertexts of $n, n-1, \dots, 2, 1, -1, -4, \dots, -(4^{m-2}), -(4^{m-1})$ in the public homomorphic encryption system. He permutes the list of ratings, such that the order does not reveal the plaintext and proves in zero knowledge that it is indeed a permutation. Efficient methods for doing so exist [15, 27]. Alternatively cut-and-choose techniques can be employed.

The second reputation service provider SP_2 , after verifying the correctness of a submission, removes all previous ratings from the rater before publishing the feedback. It can do so using his ability to link feedback. The global aggregation algorithm then proceeds as the Beta algorithm in Section 5.3.

5.3.2 Evaluation

We intend to prove by simulation that our reputation system can effectively differentiate different performers in the set of ratees. As a side effect we tune the parameters n and m of positive and negative ratings, respectively, for our experimental setup.

We emulate the difficult classification problem of Virtual Organization formation. We chose 5000 participants in the system: 1000 each with a performance rating of 0.95, 0.96, 0.97, 0.98, 0.99. A performance rating indicates the expected percentage of positive transactions.

We randomly choose a party and select its partner for the transaction according to the ranking. We sum all ratings and assign each the selection probability of its rating divided by the sum. If there are negative ratings, the minimum is added to all ratings.

The two selected partners then engage in a simulated transaction. This transaction is positive with the probability of the performance rating of the second partner (chosen by the ranking). The first partner keeps a record of all of his transactions and then leaves honest feedback using n positive and m negative ratings.

For local aggregation simple addition is used in this simulation. The more positive ratings one has, the higher its rating. The more negative ratings one has, the lower the rating. Each positive and negative transaction counts as one. The aggregated feedback is left with the reputation service provider.

We simulated a series of transactions this way and observed the overall ranking published by the reputation server. For each section of 1000 ratees we computed the average performance, e.g., the average performance of ranks 1 – 1000. Ideally, we would arrive at the different performance classes perfectly separated, but of course our simulation is randomized.

We started with a simulation of 5 positive and 5 negative ratings. The results are depicted in Figure 2. We observe that the two classes 0.98 and 0.97 performance do not get

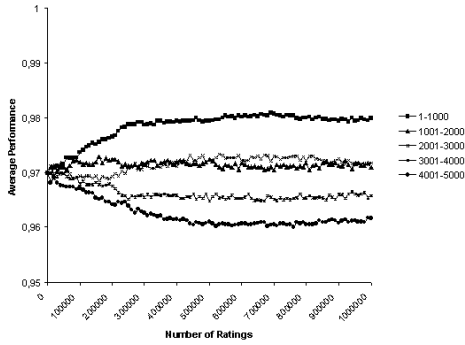


Figure 2: Average performance in the ranking with 5 positive and 5 negative ratings

separated. This indicates that there are not enough positive ratings to counterbalance the negative ones. The top performers are still rated, but the second and third fall out of the ratings and get blurred.

Although our figures only depict the results of one experiment, we repeated each experiment three times and observed very similar results each time albeit the different randomness.

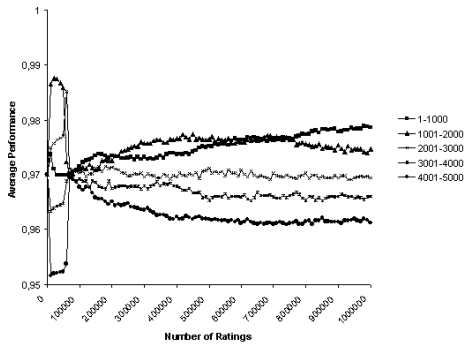


Figure 3: Average performance in the ranking with 20 positive and 5 negative ratings

We therefore increased the number of ratings to 20 positive and 5 negative ratings. The results are depicted in Figure 3. We observe a strong variation at the beginning of the experiment, but after approximately 100000 transactions the ratings stabilize. After around 800000 ratings we can see a clear separation of all performance classes in the ranking. Note that a reputation system can already be useful before rating have fully separated. We conclude that we need more positive ratings than negative ones.

In order to verify this conclusion, we set the number of positive ratings to 0. The results are depicted in Figure 4. We observe the same separation problem of the two performance classes as with 5 and 5 ratings. This supports our hypothesis that more positive ratings are necessary.

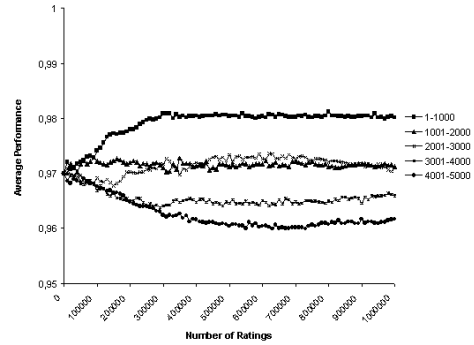


Figure 4: Average performance in the ranking with 0 positive and 5 negative ratings

6. ANONYMIZATION OF FEEDBACK

As outlined in the introduction, if every feedback left has immediate impact on the rating a rated party can deduce the feedback by observing its impact on its overall rating. This problem is aggravated, because the rated party knows when the transaction finished and the rater is able to leave feedback. Even if the rating is left entirely private, its impact is still observable.

Therefore the second service provider SP_2 has to collect multiple feedback before publishing them. The question is how many feedback SP_2 needs to collect and how long it takes to collect them.

In order to answer these questions we have investigated the MovieLens data sets¹. Although movie ratings are not necessarily comparable to ratings for Virtual Organization formation, we anticipate that the data gives some insight and rely on a best effort, since no data for VO information is available. One difference may be that there are definitely some bad movies that live forever, but we believe that such companies will not survive long in a Virtual Organization breeding environment. In this data set movies are ranked on a five-star scale from 1 to 5. Since we want to simulate the behavior of dominantly positive ratings, we consider only 1 star ratings as negative ratings and all others as positive. In the 1.000.000 ratings data set there are 5.6% 1 star ratings, which comes pretty close to our assumed negative ratings.

In order to determine the minimum number of ratings necessary to protect the individual rating we consider two approaches. In the first approach we measured the average number of ratings necessary, such that at least one negative and one positive rating is present. In the MovieLens data this number is 17.35 ratings.

The previous measure protects negative as well as positive ratings (on average), but the purpose of private ratings is to protect the negative bids. We therefore measured the maximum number of consecutively negative ratings plus one, such that each negative rating has at least one positive rating within its batch. In the MovieLens data this number is 16 ratings.

This shows a difference between the MovieLens data and the data we assume for Virtual Organization formation, since we do not assume that such long streaks of negative rat-

¹<http://www.grouplens.org/node/73>

ings might even occur. Nevertheless the relative closeness of both data points indicates that an aggregation of 15 - 20 ratings provides good privacy of ratings. Note that the service provider SP_2 has to perform this aggregation without knowledge of the rating, such that a predefined threshold is a must.

We then investigated how long it takes on average to receive k ratings. Again we feared a fundamental difference between MovieLens and our data, since movies are usually rated shortly after they appear and then rather seldom. Surprisingly this does not seem to be the case in the MovieLens data. The results of our experiment are depicted in Figure 5.

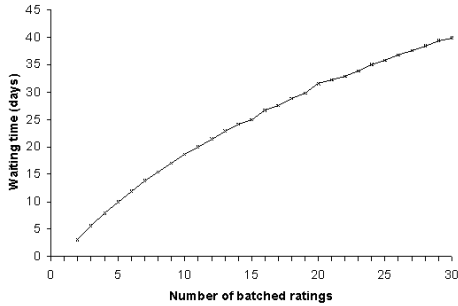


Figure 5: Average waiting time for number of ratings

We can conclude that in a reputation system with a similar user structure (number of users 6040) to the MovieLens data set the average waiting for our anonymization threshold is 25 to 31 days.

We emphasize that choosing a waiting period before publishing ratings as suggested in this section is necessary in our system and preferable to waiting for a threshold of ratings. The reason is that neither the first nor the second service provider can determine whether a threshold of ratings is new due to their view, such that neither can undoubtedly determine whether the threshold has been passed. Even worse, a malicious rater could exploit this and forge with collusion a number of additional ratings in order to pass the threshold. Keeping the waiting period ensures that with very high probability at least two raters with different ratings have left (legitimate) feedback. This protects their individual ratings from inference by the rater.

7. CONCLUSIONS

We have presented a reputation system architecture that is able to keep the feedback ratings entirely private, but does not require the transactions to be anonymous. Neither the reputation system (without collusion) nor the rater can infer the feedback. We hope to elicit honest ratings with this privacy guarantee.

In order to limit the possibility for bad-mouthing attacks we introduced a token system. The rater issues a token to the rater without which he cannot leave feedback for him. There is no need for a central registry of transactions. We proved the privacy of the token system by a reduction to the Bilinear Decisional Diffie-Hellman problem.

A further obstacle to privacy for ratings is that the rater knows when the rater is going to leave feedback and can

observe its impact on its reputation. The only solution is to combine multiple ratings, such that the rater cannot differentiate who left which feedback. We analyzed using real data how many feedback ratings need to be combined and how long it takes to collect that many ratings.

We designed a reputation score aggregation algorithm for Virtual Organization formation and showed that it is able to distinguish the complex classification problem. We can therefore conclude that our combined reputation system is capable of performing its function while providing full privacy of ratings.

Future work is to evaluate the reputation system under different strategies of the raters.

8. ACKNOWLEDGEMENTS

This work has partially been financed by the European Commission through the ICT programme under Framework 7 grant FP7-213531 to the SecureSCM project.

9. REFERENCES

- [1] R. Agrawal, and R. Srikant. Privacy-Preserving Data Mining. *ACM SIGMOD Record* 29(2), 2000.
- [2] E. Androulaki, S. Choi, S. Bellovin, and T. Malkin. Reputation Systems for Anonymous Networks. *Proceedings of the 8th International Symposium on Privacy Enhancing Technologies*, 2008.
- [3] A. Arenas, B. Aziz, and G. Silaghi. Reputation Management in Grid-Based Virtual Organisations. *Proceedings of the International Conference on Security and Cryptography*, 2008.
- [4] J. Benaloh. Verifiable Secret-Ballot Elections. *PhD thesis, Yale University*, 1987.
- [5] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. *Proceedings of the 20th ACM symposium on theory of computing*, 1988.
- [6] Y. Bo, Z. Min, and L. Guohuan. A Reputation System with Privacy and Incentive. *Proceedings of the 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, 2007.
- [7] D. Boneh, and M. Franklin. Identity Based Encryption from the Weil Pairing. *SIAM Journal of Computing* 32(3), 2003.
- [8] D. Boneh, B. Lynn, and H. Shacham. Short Signatures from the Weil Pairing. *Proceedings of Asiacrypt*, 2001.
- [9] J. Camenisch, and E. Van Herreweghen. Design and Implementation of the Idemix Anonymous Credential System. *Proceedings of the 9th ACM Conference on Computer and Communications Security*, 2002.
- [10] J. Camenisch, and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. *Proceedings of EUROCRYPT*, 2001.
- [11] J. Canny. Collaborative Filtering with Privacy. *Proceedings of the IEEE Symposium on Security and Privacy*, 2002.
- [12] T. Chen, W. Han, H. Wang, Y. Zhou, B. Xu, and B. Zang. Content Recommendation System Based on Private Dynamic User Profile. *Proceedings of the*

- International Conference on Machine Learning and Cybernetics*, 2007.
- [13] I. Damgard, and M. Jurik. A Generalisation, a Simplification and some Applications of Pailliers Probabilistic Public-Key System. *Proceedings of International Conference on Theory and Practice of Public-Key Cryptography*, 2001.
- [14] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. *Proceedings of the 19th ACM conference on theory of computing*, 1987.
- [15] J. Groth. A Verifiable Secret Shuffle of Homomorphic Encryptions. *Proceedings of the 6th International Workshop on Theory and Practice in Public Key Cryptography*, 2003.
- [16] A. Josang, R. Ismail. The Beta Reputation System. *Proceedings of the 15th Bled Electronic Commerce Conference*, 2002.
- [17] A. Joux, and K. Nguyen. Separating Decision Diffie-Hellman from Diffie-Hellman in Cryptographic Groups. IACR E-print Archive 2001/03, 2001.
- [18] F. Kerschbaum, J. Haller, Y. Karabulut, and P. Robinson. PathTrust: A Trust-Based Reputation Service for Virtual Organization Formation. *Proceedings of the 4th International Conference on Trust Management*, 2006.
- [19] M. Kinader, and S. Pearson. A Privacy-Enhanced Peer-to-Peer Reputation System. *Proceedings of the 4th International Conference on Electronic Commerce and Web Technologies*, 2003.
- [20] Y. Lindell, and B. Pinkas. Privacy Preserving Data Mining. *Proceedings of Crypto*, 2000.
- [21] A. Lysyanskaya, R. Rivest, A. Sahai, and S. Wolf. Pseudonym Systems. *Proceedings of the 6th Annual International Workshop on Selected Areas in Cryptography*, 1999.
- [22] T. Mahler, and T. Olsen. Reputation Systems and Data Protection Law. *Proceedings of e-Challenges*, 2004.
- [23] D. Naccache, and J. Stern. A New Public-Key Cryptosystem Based on Higher Residues. *Proceedings of the ACM Conference on Computer and Communications Security*, 1998.
- [24] V. Naessens, L. Demuynck, and B. De Decker. A Fair Anonymous Submission and Review System. *Proceedings of the 10th IFIP International Conference on Communications and Multimedia Security*, 2006.
- [25] A. Narayanan, and V. Shmatikov. Robust De-anonymization of Large Sparse Datasets. *Proceedings of the 29th IEEE Symposium on Security and Privacy*, 2008.
- [26] A. Narayanan, and V. Shmatikov. De-anonymizing Social Networks. *Proceedings of the 30th IEEE Symposium on Security and Privacy*, 2009.
- [27] L. Nguyen, R. Safavi-Naini, and K. Kurosawa. Verifiable shuffles: a formal model and a Paillier-based three-round construction with provable security. *International Journal of Information Security* 5(4), 2006.
- [28] T. Norman, A. Preece, S. Chalmers, N. Jennings, M. Luck, V. Dang, T. Nguyen, V. Deora, J. Shao, A. Gray, and N. Fiddian. CONOISE: Agent-based formation of virtual organisations. *Proceedings of the 23rd SGAI International Conference on Innovative Techniques and Applications of AI*, 2003.
- [29] T. Okamoto, and S. Uchiyama. A new public-key cryptosystem as secure as factoring. *Proceedings of EUROCRYPT*, 1998.
- [30] P. Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. *Proceedings of EUROCRYPT*, 1999.
- [31] E. Pavlov, J. Rosenschein, and Z. Topol. Supporting Privacy in Decentralized Additive Reputation Systems. *Proceedings of the 2nd International Conference on Trust Management*, 2004.
- [32] F. Pingel, and S. Steinbrecher. Multilateral Secure Cross-Community Reputation Systems for Internet Communities. *Proceedings of the 5th International Conference on Trust, Privacy and Security in Digital Business*, 2008.
- [33] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman. Reputation Systems. *Communications of the ACM* 43(12), 2000.
- [34] P. Robinson, F. Kerschbaum, and A. Schaad. From Business Process Choreography to Authorization Policies. *Proceedings of the 20th IFIP Conference on Data and Applications Security*, 2006.
- [35] S. Steinbrecher. Design Options for Privacy-Respecting Reputation Systems within Centralised Internet Communities. *Proceedings of the 21st IFIP International Information Security Conference*, 2006.
- [36] G. Swamynathan, B. Zhao, K. Almeroth, and R. Jammalamadaka. Towards Reliable Reputations for Dynamic Networked Systems. *Proceedings of the IEEE Symposium on Reliable Distributed Systems*, 2008.
- [37] M. Voss, A. Heinemann, M. Mühlhäuser. A Privacy Preserving Reputation System for Mobile Information Dissemination Networks. *Proceedings of the 1st International Conference on Security and Privacy for Emerging Areas in Communications Networks* 2005.
- [38] A. Yao. Protocols for Secure Computations. *Proceedings of the IEEE Symposium on foundations of computer science* 23, 1982.