Extending Signatures of Reputation*

Emmanuelle Anceaume[1], Gilles Guette[1], Paul Lajoie-Mazenc[1], Thomas Sirvent[2], and Valérie Viet Triem Tong[3]

[1] IRISA – Université de Rennes 1 – CNRS, France
firstname.lastname@irisa.fr
[2] DGA Maîtrise de l'information,
IRMAR – Université de Rennes 1, France
thomas.sirvent@m4x.org
[3] SUPELEC, France
valerie.viettriemtong@supelec.fr

Abstract. Reputation mechanisms are a powerful tool to reduce the potential risk of interacting with almost or completely unknown users in environments in which there is no incentive to behave trustworthily, e.g. in open and large-scale systems. However, by collecting feedback about users, reputation mechanisms can easily be manipulated to deduce users' profiles; thus, these mechanisms jeopardize users' privacy, which clearly compromise their wide adoption. Privacy-preserving reputation mechanisms have recently been proposed to solve this issue. All the proposed designs either rely on a trusted central authority to handle the casting of votes and the derivation of reputation scores, or are based on a distributed environment and use cryptographic tools (e.g. non-interactive zero-knowledge proofs of knowledge and homomorphic encryption) to demonstrate the validity of votes and reputation scores. However, to the best of our knowledge, all the proposed distributed mechanisms produce solely monotonic reputation scores: whatever the outcome of an interaction, a service provider's reputation can never decrease. In this article, we propose a distributed privacy-preserving reputation mechanism handling both positive and negative votes. This is achieved by combining algorithms and tools from both the distributed and the cryptographic communities.

Keywords: Distributed reputation mechanism, privacy, non-monotonic reputation score.

1 Introduction

In large scale and dynamic networks such as the Internet, most interactions occur between unknown users. When users invest time or money in such interactions, this may induce a severe risk. For instance, in e-commerce transactions, a buyer – or client – has no guarantee that the item on sale will be sent or even that

* This work has been partially supported by the French ANR project AMORES, http://amores-project.org

M. Hansen et al. (Eds.): Privacy and Identity 2014, IFIP AICT 421, pp. 165–176, 2014.

its real state is consistent with its description. Hence, there is a crucial need for clients to determine to what extent an interaction with a given user is safe. In the following, we call *service providers* the users that provide a given service, and *clients* the users that wish to obtain a given service.

Reputation mechanisms come out as an effective tool to assess this risk, and indirectly foster trust and motivate cooperation in large scale and open systems. Indeed, similarly to the word-of-mouth reputation, a reputation mechanism allows clients to form an opinion on the behavior of an unknown service provider through a *reputation score*. A reputation score is a mathematical object (e.g. a number or a percentage) computed from the set of votes cast by the past clients of the targeted service provider. Reputation scores can either be computed by a central entity – like in eBay[1] – by the users themselves [1–4] or by the targeted entity [5]. Users evaluate the risk of interacting with the targeted service provider according to his reputation score. Reputation mechanisms are thus an efficient tool to encourage service providers to trustworthily behave. On the other hand, the reputation of misbehaving service providers gradually decreases, which quickly dissuades potential clients from interacting.

In order to maintain reputation scores up to date, clients are regularly requested to send their feedback regarding their past interactions with service providers. Unfortunately, by collecting feedback from clients, reputation mechanisms can easily be manipulated to deduce users' profiles (both clients' and service providers') and thus to jeopardize their privacy. This is achieved by compromising users' *anonymity* and the *unlinkability* of their interactions [6]. Pfitzmann and Hansen's taxonomy [7] states that a user is anonymous if an attacker cannot identify him among a group of users; and that any two interactions are unlinkable if an attacker cannot tell whether they involve the same users. Privacy-preserving reputation mechanisms have recently been proposed to solve this issue. Proposed designs either rely on a trusted central authority to handle the vote casting and reputation score derivation [8], or use cryptographic tools (e.g. non-interactive zero-knowledge proofs of knowledge and homomorphic encryption) to demonstrate the validity of votes [5]. By protecting users' identities and by guaranteeing the unlinkability of their actions, Resnick et al. [9] have shown that it gives supplementary incentives for the clients to feed the reputation mechanism with honest feedback without fearing retaliation. However, to the best of our knowledge, none of the existing privacy-preserving reputation mechanisms can handle both positive and negative votes. Note that a negative vote is a vote reflecting a dissatisfied client, but not necessarily a negative vote in the mathematical sense. Negative votes allow reputation scores to exactly reflect misbehaving service providers. In particular, negative votes allow to protect the system against service providers that initially behave correctly to gain a high reputation, until they suddenly turn malicious and attract clients in fraudulent transactions. Indeed, handling both positive and negative votes is recognized as a real scientific challenge [5].

[1] `http://www.ebay.com`

In this article, we present the main principles of a reputation mechanism preserving the privacy of clients and service providers in a distributed way. This mechanism is inspired from the *signatures of reputation* proposed by Bethencourt et al. [5]. The most important feature of our proposal is the handling of both positive and negative votes. To the best of our knowledge, this is the first mechanism handling such a feature in a privacy-preserving distributed context.

The remainder of this article is organized as follows. We present the state of the art of privacy-preserving reputation mechanisms in Sect. 2, with a focus on the signatures of reputation of Bethencourt et al. [5]. In Sect. 3, we define the privacy and security properties that our mechanism guarantees. We then present the tools used in our proposal in Sect. 4, and how they are used during an interaction between a client and a provider in Sect. 5. Finally, we conclude in Sect. 6.

2 State of the Art

Pavlov, Rosenschein and Topol [10] were the first ones to propose a reputation system guaranteeing the privacy of its users in a distributed way. In contrast to subsequent works, their notion of privacy is limited to the non-disclosure of clients' votes. Androulaki et al. [8] go a step further by proposing a reputation mechanism preserving the anonymity of both service providers and clients through pseudonyms, anonymous credentials and blind signatures. On the other hand, and despite their reliance on a correct (but curious) central authority, their mechanism does not guarantee that the reputation score of a given service provider reflects its past behavior. Indeed, a service provider s_1 can easily collude with another one s_2 by giving to s_2 the *repcoins* (i.e. anonymous positive votes) he has received so that s_2 can ask the central authority to increase his own reputation score with these irregularly received repcoins. Guaranteeing the non transferability of reputation is crucial when dealing with malicious service providers. In addition, and as mentioned above, negative votes are not handled by this mechanism.

Bethencourt, Shi and Song [5] propose a cryptographic primitive called *signature of reputation*, allowing any user to advertise the services he provides with his reputation without divulging any private information. Both clients and service providers preserve their anonymity through one-time pseudonyms. The construction of signatures of reputation does not rely on any trusted central authority. Rather, each service provider computes signatures of reputation based on cryptographic votes built by his past clients. The reputation of a service provider is the count of *distinct* clients who granted him votes. This avoids ballot-stuffing attacks, i.e. attacks where a single client votes multiple times for a service provider to raise or lower his reputation. Bethencourt et al.'s system makes extensive use of Non-Interactive Zero-Knowledge proofs of knowledge (NIZK) [11]. Basically, such proofs allow to prove the knowledge of an equation solution without disclosing it. This allows both clients and service providers to prove statements such as "this pseudonym is well-formed" or "this signature of reputation is valid"

without disclosing their identity. The first drawback of their solution is that NIZKs require high computational power, bandwidth, and storage capacity. For instance, a signature of reputation computed on 100 votes takes about 50MB of storage, which is too large to be practical. The second issue is, as for all existing privacy-preserving reputation mechanisms, that reputation scores do not reflect dissatisfied clients. This is a clear impediment to the wide adoption of such privacy-preserving mechanisms. In the following, we propose a mechanism that does take into account the discontentment of clients.

3 Modelling and Main Expectations of Our Proposal

In this section, we first introduce the model and the terminology used throughout this work. Then, we present the properties that fully characterize our privacy-preserving reputation mechanism.

3.1 System Model

We consider a large scale and open system populated by clients and service providers. We assume that both clients and service providers communicate over an anonymous communication network, e.g. Tor [12]. Both clients and service providers interact via pseudonyms they generate themselves. We do not make any assumption regarding the behavior of both parties. In particular, clients may abruptly end their interaction prior to having cast their votes, and service providers may devise strategies to manipulate their own reputation, or steal the reputation of another service provider. More generally, any number of users may collude in the objective of breaking the anonymity of some other user.

3.2 Reputation Mechanism

First, we make a difference between a *transaction* and an *interaction*. A transaction represents the exchange of the service between a service provider and a client, while an interaction consists of the transaction and all the communications between the client and the service provider allowing to take into account the outcome of the transaction. Any meaningful reputation mechanism involves the following three phases. The first phase allows any service provider to prove its current reputation to any requesting client. Then, when the requirements of both parties are satisfied, the transaction takes place. Finally, the client casts a vote reflecting the quality of the transaction. Note that in order to face non-cooperative clients, that is clients disengaging from the interaction prior to having cast a vote, a mechanism attesting that a transaction did occur between both parties need to be implemented. Such proofs of interaction must appear in the reputation score of service providers.

3.3 Properties

We now present the properties that fully characterize our reputation mechanism.

Property 1 (Privacy of service providers). *The privacy of service providers is preserved if, when a client votes for an honest service provider, this service provider is anonymous among the service providers with an equivalent reputation.*

Property 2 (Privacy of clients). *The privacy of clients is preserved if:*

1. *An honest client is anonymous among all clients;*
2. *The interactions of a client with different service providers are unlinkable.*

In addition to both privacy properties, we present the properties that guarantee that no attacker can take advantage of our reputation mechanism.

Property 3 (Correctness of a reputation mechanism). *A reputation mechanism is correct if the following six properties hold.*

Unforgeability of votes *If a service provider has received n votes, then this service provider was involved in at least n transactions;*

Unforgeability of interaction proofs *If a service provider proves that n transactions occurred, then this service provider was involved in at least n transactions with distinct clients;*

Unforgeability of reputation scores *A service provider cannot prove a fake reputation score;*

Non-repudiation of votes *If, at the end of a transaction between a client and a service provider, the client casts a vote, then his vote updates the reputation score that was proved at the beginning of the interaction;*

Non-repudiation of interaction proofs *At the end of a transaction between a client and a service provider, the service provider can prove that the transaction occurred;*

Impact limitation *The impact of a single client on the reputation score of a service provider is limited.*

The unforgeability and non-repudiation properties are standard properties in reputation systems, and most systems ensure them. On the other hand, the non-repudiation of votes prevents colluding service providers from using a "good" service provider to prove his reputation – and attract clients – and then behaving maliciously and giving the negative votes to another service provider. Finally, the impact limitation property prevents ballot-stuffing attacks. Note that Bethencourt et al.'s signatures of reputation [5] guarantees those properties. However, since there are no negative votes, service providers have no interest in repudiating a vote. In the following section, we present the tools we use to guarantee all these properties.

4 Principles of Our Reputation Mechanism

Our proposal aims at enhancing the signatures of reputation proposed by Bethencourt et al. [5] by handling negative votes. Taking into account negative votes implies major modifications with respect to the implementation of the mechanism. Specifically, in Bethencourt et al.'s mechanism, service providers locally store votes cast at the end of their interaction with their clients, and compute their reputation score by aggregating the received votes. In particular, they can keep only a subset of them, which clearly makes negative votes useless. We propose to improve upon this solution by guaranteeing that negative votes are taken into account. This is achieved by making both reputation scores and votes of service providers publicly available in order to prevent anyone from modifying or hiding them. Our proposition accomplishes this without jeopardizing the privacy of clients.

4.1 Preserving the Privacy

In order to preserve the privacy of both clients and providers (Prop. 1 and 2), these entities must prove that they belong to the system without revealing their identity (in the following, we use the term *credential* as a synonymous of identity), and prove the correctness of their computations without revealing the values used in these computations. Similarly to Bethencourt et al. [5], we use Non-Interactive Zero-Knowledge proofs of knowledge (NIZK) to solve these issues.

Furthermore, the reputation scores displayed by providers are approximations of their exact reputations. Indeed, if two different pseudonyms prove that "82.476% of my clients are satisfied", it is highly plausible that those pseudonyms belong to the same provider, which breaks this service provider's anonymity. Thus, providers only prove approximations of their reputation score: showing that "between 80% and 85% of my clients are satisfied" yields roughly the same information as before, but enlarges the anonymity set of the provider.

4.2 Correctness of the Mechanism

Unforgeability of votes, interaction proofs and reputation scores. In order to guarantee the properties of unforgeability (i.e. unforgeability of votes, interaction proofs and reputation scores), we use anonymous proxy signatures [13]. They preserve the privacy of both clients and service providers due to their compatibility with NIZKs. To participate in the system, each user (i.e. clients and service providers) register with a registration authority which generates the user's credentials and certificates. With anonymous proxy signatures and NIZKs, a user can then use his certificate to prove that he sent a message or made computations without disclosing his credential among all registered clients, or service providers. Several messages or computations can be linked to a unique hidden certificate (using the same randomization of the certificate). By doing so, a client

can emit a vote for a service provider, without knowing the credential of the service provider, but with the guarantee that his vote will affect the reputation score of the service provider who proved his reputation at the beginning of the interaction.

Non-repudiation of Votes and Interaction Proofs. In the context of anonymous reputation mechanisms, both clients and service providers have antagonist expectations. Clients require that each of their votes (in particular negative ones) be taken into account, while providers expect their privacy to be preserved. However, if the provider remains anonymous when the client casts his vote, he has the opportunity to reject a negative vote simply by not revealing his identity; indeed, the vote cannot be assigned to any provider. On the other hand, if the provider reveals his identity prior to the voting phase, his privacy is broken, and the client may change both his behavior and his vote according to the provider's identity. In the same way, the provider wants to get a proof of interaction testifying that a transaction took place, while the client wishes to stay fully anonymous.

To deal with such opposite concerns, we propose to build a trusted and distributed third party through *share carriers*. At the beginning of each interaction the client and the service provider choose share carriers among the system's users so that no one controls a majority of them, even if some of the share carriers can be malicious. The role of share carriers is to obtain the identity of the service provider, prior to the vote, and disclose this identity after the vote if the service provider behaves incorrectly. The same process is used for the non-repudiation of interaction proofs. The choice of share carriers is done at the beginning of each interaction. This choice is jointly handled by both the client and the service provider that wish to engage into a given transaction. This prevents any collusions between a malicious client (or a malicious service provider) and a set of malicious share carriers. The choice of share carriers is the outcome of a nonce-based interactive protocol between the client and the service provider. To prevent some of the share carriers from disclosing any sensitive information (i.e. the identity of the provider, or the interaction proof), any information is shared using Verifiable Secret Sharing (VSS) [14]. A VSS is a (t, n)-threshold cryptographic scheme dividing a secret into n shares, such that any t shares allow any user to recompute the secret. On the other hand, less than t shares yields no information about it. Share carriers can also verify the consistency of all shares and the correctness of the secret, so that they cannot be bilked by neither the client nor the provider in the sharing process.

Impact Limitation. To guarantee the impact limitation property, we follow the strategy given by Bethencourt et al. [5] that allows to detect whether any two votes on a given service provider were cast by the same client. This detection is made through an indicator that fully characterizes a given couple (client, provider), independently of the client and provider pseudonyms. Such an indicator is called *invariant*. In Bethencourt et al.'s reputation mechanism, an

invariant is somehow a positive vote, and a service provider proves his reputation score by proving that he obtained enough distinct invariants. In our construction, the invariant is used as an interaction proof, and appears in votes, to detect multiple votes from the same client for the same provider. Note that even if the invariant is unique for a given couple, it is impossible to compute the client's credential from the invariant and the service provider's credential. Furthermore, two providers cannot distinguish whether they have interacted with the same clients or not.

In our scheme, the invariant is computed during a three steps interaction. First, the provider masks his identity, and sends the result to the client. The client then computes a masked invariant, based on the masked identity of the provider and his own identity, and sends it to the provider. Finally, the provider removes the mask, and obtains the plain invariant.

5 Protocol of Interaction

In the previous section, we have presented the tools used to guarantee both the privacy of providers and clients (Props. 1 and 2), and the mechanism's correctness (Prop. 3). We now detail the interaction protocol between a client c and a service provider p, and describe how these tools are used. An interaction is made of four successive stages.

1. Service provider p proves his reputation to client c.
2. Both c and p choose the distributed third-party for the interaction.
3. Both c and p commit themselves to the interaction by sharing a secret.
4. Once the transaction is over, they both participate to update p's reputation; there are three variants of this stage:
   (a) both c and p are correct and help each other by respectively casting a vote and getting a proof of interaction
   (b) c is malicious and refuses to help p to get a proof of the interaction,
   (c) p is malicious and refuses to help c to rate him.

We assume that, at the beginning of each interaction, p receives a certificate on his credential and his reputation.

5.1 Proof of Reputation

To prove his reputation, p sends his pseudonym nym_p and his reputation rep_p to c and proves that he knows a certificate on his reputation thanks to a NIZK. Namely, p demonstrates that he knows

- cred_p such that nym_p was issued from cred_p, and
- cert_p certifying $\langle \mathrm{cred}_p, \mathrm{rep}_p \rangle$

Once c has checked the proof and if he is comfortable with the reputation score of p, then c engages an interaction with p. Note that all the proofs (i.e. reputations and computations) are done through NIZKs.

5.2 Choosing the Share Carriers

Both c and p need to choose the share carriers they will rely on throughout their interaction as presented in Section 4.2. The number n of chosen share carriers depends on two system parameters – the total number N of user, and the proportion m of malicious users – and on the maximal probability P of having a collusion among the share carriers, that is of having more than $n/2$ malicious share carriers. Table 1 shows n for multiple values of N and m, for $P = 2^{-80}$.

Table 1. Required number of share carriers to prevent collusions as a function of N (total number of share carriers) and m (percentage of malicious share carriers)

m \ N	100	500	1,000	5,000	10,000
5%	9	45	53	61	61
6%	11	51	59	69	69
7%	13	55	67	75	77
8%	15	61	73	83	85
9%	17	67	79	93	93
10%	19	73	87	101	103

To choose the share carriers, both c and p use a hash function H (e.g. SHA-256 [15]), with h-bits outputs, and proceed as follows.

1. c chooses a random nonce r_c and commits himself to r_c by sending $H(\texttt{00} \,\|\, r_c)$ to p;
2. p chooses a random nonce r_p and sends it to c;
3. c sends r_c to p;

The share carriers are the $\left\{\lfloor H(\texttt{01} \,\|\, r_p \,\|\, r_c \,\|\, i) \times N/2^h \rfloor, i \in \{0, \ldots, n' - 1\}\right\}$, where all users are labelled $0, \ldots, N - 1$ and n' is set so that the set contains exactly n distinct share carriers.

5.3 Completion of a Transaction

Once both client c and service provider p have agreed on the set of share carriers, c must make sure that p will allow him to cast a vote whatever the issue of the transaction. Similarly p must be guaranteed that c will allow him to get a proof of interaction whatever the issue of the transaction. In both cases this is achieved by having c and p commit to secrets, which they share among the share carriers. From Sect. 4.2, computation of the invariant requires that both c and p be involved in an interaction, we use it as the proof of interaction between c and p. Thus, the secret of c is the masked invariant while the one of p is its credential cred_p. Hence, even if either c or p disconnect after the transaction, the other one will be able to reconstruct either p's identity for c

or the invariant, that is the proof of interaction, for p. Once computed, p sends his masked credential to c, c computes the masked invariant and both share their secret among the share carriers. Suppose that client c wants to split a secret $x \in \mathbb{Z}_k$ into n shares and allows its reconstruction with any t of them. Then, c randomly chooses $a_j \in \mathbb{Z}_k, j \in \{1, \ldots, t-1\}$, defines $a_0 = x$, and $q : z \mapsto \sum_{j=0}^{t-1} a_j z^j$. Client c sends the i-th share $q_i = q(i)$ to the i-th share carrier. Client c also computes commitments C_{a_j}, C_{q_i} to each a_j and q_i with a homomorphic commitment scheme (such as the SXDH commitments [11]). The main property of such commitments is that if C_{z_1}, C_{z_2} are commitments to $z_1, z_2 \in \mathbb{Z}$, then $C_{z_1} \cdot C_{z_2}^{\lambda}$ is a commitment to $z_1 + \lambda z_2$. For each share, c also computes two NIZKs: the first one is a proof that C_{q_i} is a commitment to q_i. The second one is a proof that all the share are consistent, i.e. a proof that $C_{q_i} \cdot \prod_{j=0}^{t-1} (C_{a_j})^{-(i^j)}$ is a commitment to 0; indeed, thanks to the homomorphic and commitment properties, this proof demonstrates that

$$q_i = \sum_{j=0}^{t-1} a_j \, (i^j) = q(i).$$

Once the secrets have been shared and verified, c and p can proceed with the transaction. Figure 1 illustrates the interactions described in Sect. 5.2 and 5.3.

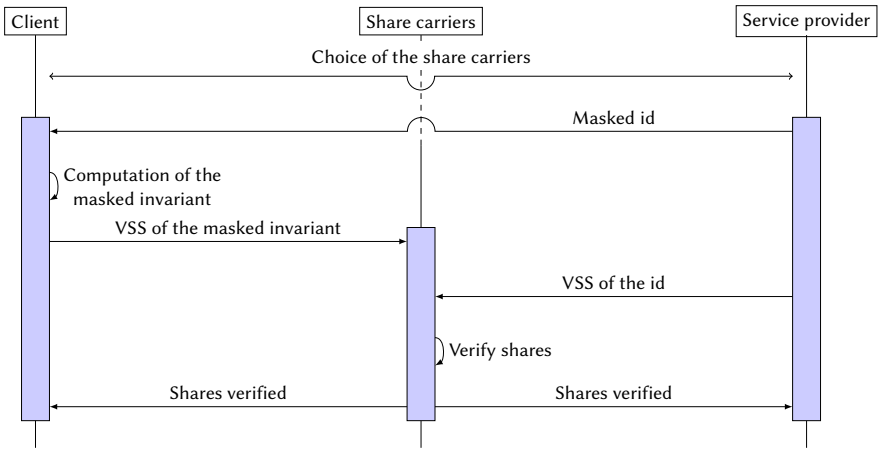

Fig. 1. Preparation phase of the transaction between a client and a service provider

5.4 Casting a Vote

Once both c and p have finished their transaction, the reputation of p can be updated thanks to c's vote (if any), or in the negative thanks to the proof of interaction. There are three different outcomes of the protocol according to both c and p behaviors, that is, depending on whether they are willing to give their secrets.

Suppose that both behave correctly. Then c simply gives his vote and the masked invariant to the share carriers, who transmit the masked invariant to p. Note that sending the masked invariant to the share carriers save them from recomputing it from their shares. Then, p reveals his credential to c and computes the invariant. Afterwards, the vote is cast on p and p's reputation is updated with c's vote.

Now assume that p refuses to reveal his credential. Then the share carriers give p's shares to c once c has cast his vote, allowing c to obtain p's identity. Thus, c is able to compute the invariant and to emit the vote.

Finally suppose that c refuses to give the masked invariant to p. Then whether c has given his vote to the share carriers or not, the share carriers give c's shares to p, which allows p to compute the invariant and obtain the proof of interaction. Figure 2 shows the vote emission when both c and p are correct.

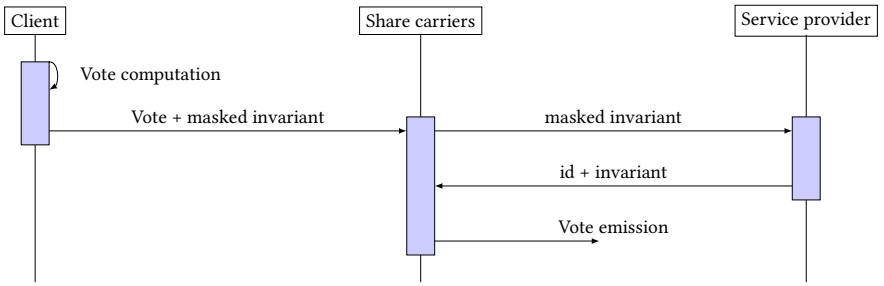

Fig. 2. Vote emission

6 Conclusion

In this article, we have presented a reputation mechanism addressing two main issues of reputation mechanisms: preserving all users' privacy *and* computing reputation scores based on both positive and negative ratings. There were already systems addressing those two issues, but, to our knowledge, none addressed both of them at the same time. We achieve this thanks to cryptographic schemes such as zero-knowledge proofs, verifiable secret sharing, and anonymous proxy signatures. Furthermore, our proposition is independent of the reputation model; that is, our system can integrate any reputation model [1,4], preferably one using both positive and negative ratings, and limiting the impact of a single client on a given provider.

Our proposal works in a distributed fashion: no trusted central authority is required for either the correctness or the users' privacy. However, this requires many certifications done by all the possible share carriers, which might be too expensive for a large-scale system. For future work, we intend to quantify those costs for an implementation. If they are too high, we will study to what extent a central authority could lower these costs while preserving the users' privacy.

References

1. Jøsang, A., Ismail, R.: The beta reputation system. In: Proceedings of the 15th Bled Electronic Commerce Conference, pp. 41–55 (2002)
2. Yu, B., Singh, M.P.: Distributed reputation management for electronic commerce. Computational Intelligence 18, 535–549 (2002)
3. Anceaume, E., Ravoaja, A.: Incentive-based robust reputation mechanism for p2p services. In: Shvartsman, M.M.A.A. (ed.) OPODIS 2006. LNCS, vol. 4305, pp. 305–319. Springer, Heidelberg (2006)
4. Anceaume, E., Guette, G., Lajoie-Mazenc, P., Prigent, N.: Viet Triem Tong, V.: A privacy preserving distributed reputation mechanism. In: Proceedings of the IEEE International Conference on Communications (ICC) (2013)
5. Bethencourt, J., Shi, E., Song, D.: Signatures of reputation. In: Sion, R. (ed.) FC 2010. LNCS, vol. 6052, pp. 400–407. Springer, Heidelberg (2010)
6. Steinbrecher, S.: Enhancing multilateral security in and by reputation systems. In: Matyáš, V., Fischer-Hübner, S., Cvrček, D., Švenda, P. (eds.) The Future of Identity. IFIP AICT, vol. 298, pp. 135–150. Springer, Heidelberg (2009)
7. Pfitzmann, A., Hansen, M.: A terminology for talking about privacy by data minimization, v0.34 (2010)
8. Androulaki, E., Choi, S.G., Bellovin, S.M., Malkin, T.: Reputation systems for anonymous networks. In: Borisov, N., Goldberg, I. (eds.) PETS 2008. LNCS, vol. 5134, pp. 202–218. Springer, Heidelberg (2008)
9. Resnick, P., Zeckhauser, R.: Trust among strangers in internet transactions: Empirical analysis of ebay's reputation system. Advances in Applied Microeconomics 11, 127–157 (2002)
10. Pavlov, E., Rosenschein, J.S., Topol, Z.: Supporting privacy in decentralized additive reputation systems. In: Jensen, C., Poslad, S., Dimitrakos, T. (eds.) iTrust 2004. LNCS, vol. 2995, pp. 108–119. Springer, Heidelberg (2004)
11. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008)
12. Dingledine, R., Mathewson, N., Syverson, P.F.: Tor: The second-generation onion router. In: USENIX Security Symposium, pp. 303–320. USENIX (2004)
13. Fuchsbauer, G., Pointcheval, D.: Anonymous proxy signatures. In: Ostrovsky, R., De Prisco, R., Visconti, I. (eds.) SCN 2008. LNCS, vol. 5229, pp. 201–217. Springer, Heidelberg (2008)
14. Feldman, P.: A practical scheme for non-interactive verifiable secret sharing. In: FOCS, pp. 427–437. IEEE Computer Society (1987)
15. National Institute of Standards and Technology: Secure hash standard (SHS). Technical Report FIPS 180-4 (2011)