Efficient Distributed Privacy-Preserving Reputation Mechanism Handling Non-Monotonic Ratings

Paul Lajoie-Mazenc, Emmanuelle Anceaume, Gilles Guette, Thomas Sirvent, Valérie Viet Triem Tong

Efficient Distributed Privacy-Preserving Reputation Mechanism Handling Non-Monotonic Ratings

Paul Lajoie-Mazenc*, Emmanuelle Anceaume†, Gilles Guette*, Thomas Sirvent‡, and Valérie Viet Triem Tong§

*Université de Rennes 1/IRISA,
Rennes, France,
firstname.lastname@irisa.fr

†CNRS/IRISA,
Rennes, France,
emmanuelle.anceaume@irisa.fr

‡DGA Maîtrise de l'Information/IRISA,
Rennes, France,
thomas.sirvent@m4x.org

§CentraleSupélec/IRISA,
Rennes, France,
valerie.viettriemtong@supelec.fr

***Abstract*—Open and large-scale systems do not encourage their users to behave trustworthily, which may entail non-negligible risks when interacting with unknown users, for instance when buying an item on an e-commerce platform. Reputation mechanisms reduce these risks by associating a reputation score to each user, summarizing their past behavior. To be useful to users, reputation mechanisms need to guarantee two main properties: the non-monotonicity of reputation scores, in order to exactly reflect the users' behavior, and the privacy of their users, so that the history of their transactions is not publicly available. We propose a distributed privacy-preserving reputation mechanism handling non-monotonic ratings. Our proposition relies on two distinct distributed third parties and on cryptographic tools, including zero-knowledge proofs of knowledge, anonymous proxy signatures, and verifiable secret sharing. We show that this proposal is computationally efficient, and thus practical. To the best of our knowledge, this solution is the first one that preserves users' privacy and handles both positive and negative ratings without relying on a central authority.**

***Index Terms*—Reputation Mechanisms, Security, Privacy, Distributed Systems**

I. Introduction

Reputation mechanisms have come out as an effective tool to encourage trust and cooperation in electronic environments, *e.g.* e-commerce applications or web-based communities. A reputation mechanism enables users to rate services or people based on their experience [18]. These ratings or feedback are aggregated to derive reputation scores. Scores are publicly available, which allows any user to evaluate the trustworthiness of target entities (the ratees). Thus, reputation mechanisms identify capable entities and discourage others from behaving incorrectly [12]. Some reputation mechanisms require a central authority to correctly process ratings and compute reputation scores, while others rely on distributed components. The former design introduces a single point of vulnerability, allowing a single entity to fully control all reputation scores. As will be shown in the following, our reputation mechanism meets security and trust exigences through distributed computations.

While aggregating ratings is necessary to derive reputation scores, identifiers and ratings are personal data, whose collect and use may fall under legislation [14]. Furthermore, as shown by recent works [27], solely relying on pseudonyms to interact is not sufficient to guarantee user privacy [32]. This has given rise to the proposition of a series of reputation mechanisms which address either the non-exposure of the history of raters [7], the non-disclosure of individual feedback [19], [24], [29], the secrecy of ratings and the k-anonymity of ratees [11], or the anonymity and unlinkability of both raters and ratees [4], [7].

Regrettably, the search for privacy has led to restrictions in the computation of the reputation score: clients cannot issue negative ratings anymore [4], [7]. This restriction comes from the management of ratings. In existing privacy-preserving mechanisms, the ratees have the opportunity to skip some of the received ratings to increase their privacy. This is not conceivable in a non-monotonic reputation mechanism: ratees could skip negative ratings to increase their reputation scores. This is a problem because a ratee that received a thousand positive ratings and no negative ratings is indistinguishable from a ratee that received a thousand positive ratings and a thousand negative ratings. This clearly does not encourage ratees to behave correctly. Furthermore, Baumeister *et al.*

explain that "bad feedback has stronger effects than good feedback" on our opinions [6]. Negative ratings are thus essential to reputation mechanisms. So far, preserving the privacy of both raters and ratees and handling both positive and negative ratings has been recognized as a complex challenge. Quoting Bethencourt *et al.*, "Most importantly, how can we support non-monotonic reputation systems, which can express and enforce bad reputation as well as good? Answering this question will require innovative definitions as well as cryptographic constructions" [7].

The design of privacy-preserving non-monotonic reputation mechanisms is highly desirable in many applications, where both quality of service and privacy are very important. This is particularly true in all audit processes, and web-based community applications. As an example, let us focus on an application concerning all of us, the scientific peer review process.

This process is at the core of many scientific committees, including those of conferences, journals, and grant applications. Its goal is to assess the quality of research through experts, the peer reviewers, that evaluate manuscripts and proposals based on their scientific quality, significance, and originality. Peer reviewing is an activity that requires to spend considerable effort and time for doing legitimate, rigorous and ethical reviews. Reviewers are chosen by committee members among all their colleagues and peers. In order to remove any bias, several reviewers are assigned to each manuscript, and the reviews are double-blind. The increasing number of solicitations makes the reviewing task even more challenging and, by force of circumstances, may lower the quality of reviews.

Imagine now a privacy-preserving reputation mechanism whose goal would be to assess reviews according to their helpfulness and fairness. Authors would anonymously rate each received review according to both criteria. Journal editors or committee chairs would collect these ratings to update the reputation score of the concerned reviewers. Those reputations would be maintained in a very large shared anonymous repository organized according to the thematic of the anonymized reviewers. Editors and chairs would then have the opportunity to choose anonymous reviewers from this repository according to their reputation to build their reviewing committee. The rationale of such a repository is three-fold. Authors would have an incentive to honestly and carefully rate each received review as their goal would be to contribute to the creation of a pool of helpful and fair reviewers. Committee chairs would take advantage of using such a repository as they could decrease the load imposed to each selected reviewer by soliciting a very large number of them. Consequently, each reviewer could devote more time to their few reviews, and would thus provide highly helpful reviews, increasing accordingly the quality of published articles or granted applications. Finally, it would be at reviewers' advantage to provide helpful reviews to have a high reputation, which would be a unanimous acknowledgment of their expertise. As a consequence, such an anonymous repository would increase the quality of accepted manuscripts and proposals in a fully privacy-preserving way.

In the remaining of the paper, we present the design and evaluation of a non-monotonic distributed reputation mechanism preserving the privacy of both parties. After having presented the state of art in Section II, we motivate and present the properties that should be met by a reputation mechanism to be secure, to respect the privacy of all its parties, and to handle non-monotonic ratings in Section III. Section IV then provides a description of the main principles of our approach to build such a mechanism. As detailed in Section V, it relies on two distinct distributed third-parties and cryptographic tools. The orchestration of these tools is presented in Section VI. We finally show in Section VII that this unprecedented mechanism is computationally efficient, and thus implementable in large-scale applications. Section VIII concludes.

II. State of the Art

One of the first examples of reputation mechanisms has been set up by eBay. In this mechanism, clients and service providers rate each other after each transaction: ratings are either $+1$, 0, or -1 according to the (dis)satisfaction of users. The reputation score of a user is simply the sum of the received ratings. Resnick and Zeckhauser have analyzed this mechanism and the effects of reputation on eBay [31], and have highlighted a strong bias toward positive ratings. More elaborated reputations mechanisms have been proposed, such as the Beta Reputation System [21], methods based on the Dempster-Shafer theory of belief [34], or using Distributed Hash Tables to collect ratings or manage reputation scores [3], [20], [23]. Jøsang *et al.* propose a broad survey of reputation mechanisms and reputation score functions [22], while Marti and Garcia-Molina focus on their implementation in P2P systems [26]. Indubitably, the nature of ratings and the computation of reputation scores have been thoroughly researched. In this work, we do not make any assumptions regarding the function that computes reputation scores. Indeed, our solution handles both positive and negative ratings, and may thus use any computation function.

One of the first known reputation mechanism taking the privacy of users into account has been proposed by Pavlov *et al.* [29]. Their solution presents a series of distributed algorithms for computing the reputation score of service providers without divulging the ratings issued by clients. Their solution has been improved by Hasan *et al.* [19], [20] for different adversary models, and stronger privacy guarantees. Similarly, Kerschbaum proposes a centralized mechanism computing the reputation scores of service providers, without disclosing the individual ratings of the clients [24].

The secrecy of ratings contributes to the privacy of users, but is clearly insufficient: service providers can still discriminate their clients according to their identity or to additional information unrelated to the transaction. As we previously mentioned, identifiers and ratings are data that can be considered personal. Specifically, as pointed out by Mahler and Olsen [25], the European Directive 95/46/EC on data protection states that "the data [must not] be excessive in relation

to the purposes for which they are processed" [14]. A priori, the identity of users have no relation to their behavior. Thus, identities have no added utility for reputation mechanisms and are not necessary. Steinbrecher argues that reputation mechanisms must guarantee both the anonymity of their users, and the unlinkability of their transactions to be fully adopted [32]. Both properties have been lately formalized by Pfitzmann and Hansen [30]. Namely, a user is *anonymous* if this user is not identifiable within a set of users, called the *anonymity set*. The transactions of a user are *unlinkable* if the participants in two different transactions cannot be distinguished.

Hence, Clauß *et al.* [11] propose a centralized mechanism guaranteeing both the secrecy of ratings and the k-anonymity of service providers. However, this mechanism does not preserve the privacy of clients. Androulaki *et al.* [4] propose a centralized reputation mechanism guaranteeing both the anonymity and unlinkability of both parties. However, since providers send a request to the central bank for their ratings to be taken into account, only positive ratings are handled. In addition, this mechanism is vulnerable to *ballot-stuffing* attacks [12], that is, a single client can issue many ratings on a provider to bias her reputation. Hence, a provider with thousands of positive ratings from a single client is indistinguishable from a provider with positive ratings from thousands of different users, which is problematic.

Whitby *et al.* [33] propose a technique mitigating ballot-stuffing attacks, however, such a technique requires the ability to link the ratings concerning the same provider. Bethencourt *et al.* [7] propose to compute such a link. That is, they propose a mechanism linking all the transactions that have occurred with the same partners (client and service provider), while preserving their privacy. However, their reputation mechanism requires high computational power, bandwidth and storage capacity. For instance, when proving their reputation score, providers must send about 500 kB per received rating, which is from a practical point of view unbearable. Furthermore, clients cannot issue negative ratings.

We are not aware of any reputation mechanism preserving the privacy of its users and allowing clients to issue negative ratings. So far, achieving both at the same time has been recognized as a complex challenge. This is the objective of this paper.

III. Model and Properties

A. Terminology

In the following, we differentiate transactions from interactions. More precisely, a *transaction* corresponds to the exchange of a service between a client and a service provider. An *interaction* corresponds to the whole protocol followed by the client and the provider, during which the clients gets the provider's reputation and the client issues a rating on the provider. Note that we make no assumption about the nature of transactions: they can be reviews, web-based community applications, or the purchase and delivery of physical goods.

Once a transaction is over, the client is expected to issue a rating representative of the provider's behavior during the transaction. Nevertheless, clients can omit to issue such a rating, deliberately or not. While dissatisfied clients almost always issue a rating, this is not the case of satisfied clients. To cope with this asymmetry, we introduce the notion of *proofs of transaction*: a proof of transaction is a token delivered to providers for transactions when the client did not issue a rating. Such proofs of transaction allow clients to distinguish between multiple providers that have the same reputation. We denote by *report* the proof of transaction associated with the client's rating, if any. These reports serve as the basis to compute reputation scores.

Finally, we say that a user is *honest* if this user follows the protocol of the reputation mechanism. Otherwise, this user is *malicious*. Finally, a report or a reputation score is *valid* if an honest user accepts it. A more rigorous definition is given in Section VI.

B. Model of the System

We consider an open system populated by a large number of users, who can be malicious. Before entering the system, users register to a central authority $\mathcal{C}$, that gives them identifiers and certificates. Once registered, users do not need to interact with $\mathcal{C}$ anymore. A user can act as a client, as a service provider, or as both, and obtains credentials for both roles. We assume that users communicate over an anonymous communication network, *e.g.* Tor [13], to prevent the tracking of their IP addresses.

C. Properties of Reputation Mechanisms

Our reputation mechanism aims at offering three main guarantees to users. First and foremost, the privacy of users must be preserved. Second, the issuing of reports must take place without any problems. Finally, every data needed for the computation of reputation scores must be available and unfalsifiable. Note that we only give intuitive views of the properties; we defer their formal statements and their proofs in Appendix B.

Privacy properties are detailed by Property 1 and 2. Property 1 stipulates that clients do not know the service providers they interacted with when they rate them. Property 2 claims that any two clients have to remain indistinguishable. Properties 3 and 4 are related to the undeniability of reports. These properties expect that providers obtain proofs of transaction, and that clients are able to issue ratings. Property 5 tackles with the unforgeability of reports. Finally, Properties 6 and 7 respectively stipulate that computation of the reputation scores cannot be biased by ballot-stuffing attacks, and that reputation scores are unforgeable.

Property 1: Privacy of service providers. When a client rates an honest service provider, this service provider is anonymous among all honest service providers with an equivalent reputation.

Property 2: Privacy of clients. When a service provider conducts a transaction with an honest client, this client is anonymous among all honest clients. Furthermore, the interactions of honest clients with different service providers are unlinkable.

Property 3: Undeniability of ratings. At the end of a transaction between a client and a service provider, the client can issue a valid rating, which will be taken into account in the reputation score of the provider.

Property 4: Undeniability of proofs of transaction. At the end of a transaction between a client and a service provider, the provider can obtain a valid proof of transaction.

Property 5: Unforgeability of reports. Let r be a report involving a client and a service provider. If r is valid and either the client or the provider is honest, then r was issued at the end of an interaction between both users.

Property 6: Linkability of reports. Two valid reports emitted by the same client on the same service provider are publicly linkable.

Property 7: Unforgeability of reputation scores. A service provider cannot forge a valid reputation score different from the one computed from all her reports.

IV. Principles of the solution

A. Distributed Trusted Third-Parties

We explained in Section I that service providers must not manage themselves their reputation score to guarantee their reliability. To solve this issue, we propose to construct a distributed trusted authority in charge of updating and certifying reputation scores. We call *accredited signers* the entities constituting this authority. This first distributed authority has two main features. Firstly, this authority must involve fairly trusted entities or enough entities to guarantee that the malicious behavior of some of them never compromises the computation of reputation scores. Secondly, this authority must ensure that providers remain indistinguishable from each others.

Moreover, to ensure the undeniability of ratings (Property 3), a client must be able to issue his report, even if the service provider does not complete the interaction. However, the precautions taken for that purpose must not imply sending identifying data before the transaction. In the same way, data identifying the client must not be sent before the transaction to ensure the undeniability of proof of transactions (Property 4).

To solve this issue, we propose a distributed trusted authority in charge of guaranteeing that reports can be built. This distributed authority must collect information before the transaction, and potentially help one of the two parties afterwards: it must thus be online. We call *share carriers* the entities constituting this authority.

Both distributed authorities could be gathered in a single one. The drawback of this approach is that this distributed trusted authority should be simultaneously online, unique, and fairly trusted or reasonably large. The uniqueness and the participation in each interaction would induce an excessive load on each entity of this distributed authority. We thus suggest distinct authorities, for efficiency reasons. Accredited signers are then a unique set of fairly trusted or numerous entities, periodically updating the reputation scores of all providers. On the other hand, share carriers are chosen dynamically during each interaction among all service providers. Accredited signers manage every reputation score, and are thus critical in our mechanism. On the other side, share carriers are responsible for the issuing of a single report. Hence, they do not need to be as trustworthy as the accredited signers.

To deal with the privacy of both clients and providers, share carriers use *verifiable secret sharing* [15]. This basically consists in disseminating shares of a secret to the share carriers, so that they cannot individually recover the secret, but allow the collaborative reconstruction of this secret.

B. Sketch of the Protocol

The ultimate goal of our solution is to provide clients with an access to a shared repository, in which a list of services are recorded together with the reputation scores of the anonymous providers that supply those services. Clients can select services based on the displayed reputation score, which automatically sends an anonymous invitation to the associated providers. The service provider replies by sending her pseudonym, and proofs that both the pseudonym and the displayed reputation are valid. Since providers and clients must not reveal any personal information, they both use *zero-knowledge proofs of knowledge* [17] to prove the validity of their pseudonyms. Note that both clients and providers compute their own pseudonyms, which they are free to renew at each interaction. Providers and clients also need to sign their messages to guarantee their integrity, without revealing their identity. They thus use *anonymous proxy signatures* [1]. Meanwhile, they agree on the set of share carriers they will rely on.

The next step of the protocol guarantees the undeniability of ratings and proofs of transaction. As aforementioned, a provider uses verifiable secret sharing to split her identifier among all share carriers. Thanks to the zero-knowledge proof system, the provider is also able to prove that the secret was correctly shared. Similarly, the client uses verifiable secret sharing and zero-knowledge proofs of knowledge to guarantee that the provider will obtain a proof of transaction. Once this step is over, both parties engage in their transaction.

At the end of the transaction, both parties issue their report through an interactive process. As will be detailed in the following, a report contains the rating of the client as well as the identifier of the provider. Note that the presence of the provider identifier does not violate Property 1 since the rating of the client is signed before the service provider reveals her identifier. If the client does not want to rate the service, then the provider obtains a proof of transaction from the share carriers. On the other hand, if the provider does not want to help in the construction of the report, then the client signs the rating and obtains the identifier of the provider from the share carriers.

Periodically, the accredited signers gather all the issued reports, and verify them. They then update and sign the reputation scores of all service providers. Thanks to these signatures, service providers will be able to prove their updated reputation score during their subsequent interactions. More details are given in Sections V and VI.

V. Building Blocks

This section describes the tools used during interactions to guarantee both privacy and security of the users.

A. Building Trust

1) Share Carriers: As previously explained, at the beginning of the interaction, both the client and the provider randomly choose sufficiently many users to constitute the share carriers. Specifically, both parties independently draw their own nonces, *i.e.* random numbers, and combine them to deterministically select the share carriers. The independent choice of the nonces guarantees that neither the client nor the service provider are able to choose the share carriers by themselves. Let seed be the combination of the drawn nonces, $\{0, 1, \ldots, N-1\}$ be the set of potential share carriers (*i.e.* the set of service providers), and n_{SC} be the number of share carriers to be selected. The function $\mathrm{ChooseSC}(\mathsf{seed}, N)$ returns a set of n_{SC} share carriers randomly chosen in the system. We have

$$\mathrm{ChooseSC}(\mathsf{seed}, N) = \left\{ \left\lfloor H(\mathtt{01} \parallel \mathsf{seed} \parallel i) \times \frac{N}{2^h} \right\rfloor, i \in \{0, \ldots, n' - 1\} \right\},$$

where H is a hash function, *e.g.* SHA-256 [28], with h-bits output and n' is chosen so that the resulting set contains n_{SC} different share carriers. The idea behind this function is to divide the set $\{0, 1, \ldots, 2^h - 1\}$ into N intervals of similar sizes, and to compute a sequence of pseudo-random elements in $\{0, 1, \ldots, 2^h - 1\}$ from the seed. Each element points out an interval, and the first n_{SC} different intervals corresponding to the first elements in the sequence are selected. We obtain the n_{SC} corresponding share carriers. Appendix A discusses the number of share carriers that must be chosen to prevent collusions.

2) Accredited Signers: Accredited signers are users chosen in the system. There are no restrictive requisites for the selection of these users within the system, but choosing them among reasonably available users will have less impact on the latency of the reputation score updates. For instance, the central authority $\mathcal{C}$ (see Section III-B) can be in charge of choosing the accredited signers. These accredited signers obtain from $\mathcal{C}$ a delegation for the verification of reports, and the signature of reputation scores. In the following, n_{AS} represents the number of accredited signers. We assume that at least a majority of the accredited signers are honest.

As previously described, the aggregation of the reports and the computation of reputation scores is done off-line, without any direct impact on the interactions. Therefore, the accredited signers do not sign reputation scores after each interaction. Rather, they sign scores at fixed intervals of time, that we call *rounds* in the following. Both the duration and start time of each round are public parameters. A reasonable trade-off is to sign reports every day. To prove their reputation score, service providers use signatures on the current round from a majority of the accredited signers, *i.e.* at least $t_{\mathrm{AS}} = (n_{\mathrm{AS}} + 1)/2$ signatures.

B. Cryptographic Tools

We now present the cryptographic tools used by our reputation protocol, that is, the invariant to prevent ballot stuffing; the non-interactive zero-knowledge proof system; an anonymous proxy signature scheme that preserves the privacy of users; finally, a verifiable secret sharing scheme that guarantees the undeniability of reports.

1) Setting: The underlying structure for our cryptographic tools is a bilinear group $\Lambda = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G_1, G_2)$ in which $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are three groups of prime order p that we write multiplicatively. The map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is non-degenerate and bilinear. $G_1 \in \mathbb{G}_1$ (and resp. $G_2 \in \mathbb{G}_2$) is a group generator of $\mathbb{G}_1$ (resp. $\mathbb{G}_2$). The security of the presented tools relies on the intractability of the Symmetric eXternal Diffie-Hellman (SXDH) assumption in Λ, and on a specific assumption along the same lines as the Strong Diffie Hellman (SDH) assumption [1]. In the following, if X is a tuple then $X_{\mathbf{k}}$ represents the $\mathbf{k}$-th term of X.

2) Invariant: As detailed in Section II, reputation mechanisms are potential targets of ballot-stuffing attacks, in which a single client rates a service provider multiple times, in order to heavily bias the reputation of the provider. We protect our system against such attacks by strongly linking each interaction through an *invariant* jointly computed by both interacting parties. Note that Bethencourt *et al.* proposed a similar approach, though more complex [7]. We build the invariant as follows. It is derived from the identifiers of both parties, namely $\mathsf{Id}_{\mathrm{SP}} \in \mathbb{G}_1$ for the service provider, and $\mathsf{id}_{\mathrm{Cl}} \in \mathbb{Z}_p$ for the client. We define the invariant as

$$\mathrm{Inv}(\mathsf{Id}_{\mathrm{SP}}, \mathsf{id}_{\mathrm{Cl}}) = (\mathsf{Id}_{\mathrm{SP}})^{\mathsf{id}_{\mathrm{Cl}}}.$$

To preserve the privacy of both interacting parties, the invariant cannot be directly computed before the transaction. Indeed, in order to compute it directly, a user must know the identifiers of both the provider and the client. Thus, we propose to compute the invariant in three steps, which require a fixed group element $Y_1 \in \mathbb{G}_1$ randomly generated. First, the service provider computes a *pre-invariant* by masking her identifier $\mathsf{Id}_{\mathrm{SP}}$ with randomness $r \in \mathbb{Z}_p$ (G_1 and Y_1 are public elements). We have

$$\mathsf{pre\_inv} = \mathrm{Pre\_inv}(\mathsf{Id}_{\mathrm{SP}}, r) = ({G_1}^r, \mathsf{Id}_{\mathrm{SP}} \cdot {Y_1}^r).$$

Then, the client injects his identifier $\mathsf{id}_{\mathrm{Cl}} \in \mathbb{Z}_p$ and randomness $s \in \mathbb{Z}_p$ in the previous result to compute a *masked invariant* defined as

$$\begin{aligned} \mathsf{masked\_inv} &= \mathrm{Mask}(\mathsf{pre\_inv}, \mathsf{id}_{\mathrm{Cl}}, s) \\ &= ({G_1}^s \cdot {Y_1}^{\mathsf{id}_{\mathrm{Cl}}}, \mathsf{pre\_inv}_{\mathbf{1}}{}^s \cdot \mathsf{pre\_inv}_{\mathbf{2}}{}^{\mathsf{id}_{\mathrm{Cl}}}) \\ &= \left({G_1}^s \cdot {Y_1}^{\mathsf{id}_{\mathrm{Cl}}}, \mathsf{Id}_{\mathrm{SP}}{}^{\mathsf{id}_{\mathrm{Cl}}} \cdot ({G_1}^s \cdot {Y_1}^{\mathsf{id}_{\mathrm{Cl}}})^r\right). \end{aligned}$$

Finally, the provider obtains the invariant from $\mathsf{masked\_inv}$ by computing

$$\begin{aligned} \mathrm{Unmask}(\mathsf{masked\_inv}, r) &= \mathsf{masked\_inv}_{\mathbf{2}} \cdot \mathsf{masked\_inv}_{\mathbf{1}}{}^{-r} \\ &= (\mathsf{Id}_{\mathrm{SP}})^{\mathsf{id}_{\mathrm{Cl}}} = \mathrm{Inv}(\mathsf{Id}_{\mathrm{SP}}, \mathsf{id}_{\mathrm{Cl}}). \end{aligned}$$

3) SXDH Commitments: SXDH commitments [17] in a multiplicative group $\mathbb{G}_1$ (resp. $\mathbb{G}_2$) generated by G_1 (resp. G_2) permit to commit to values X in this group without revealing them. Note that it is also possible to commit to scalars $m \in \mathbb{Z}_p$, by taking $X = {G_1}^m$.

Committing to $X \in \mathbb{G}_1$ requires a commitment key (a tuple of group elements), and two random elements r, $r' \in \mathbb{Z}_p$. The commitment is denoted by $C_X = \mathrm{Com}\big(X, (r, r')\big)$. Opening a commitment C requires an opening key, that is a scalar α associated to the commitment key. The opening is denoted by $X = \mathrm{Open}(C_X, \alpha)$.

In the remainder of the paper, whenever we write that x is the value committed *in* C_x, we mean that C_x is a commitment to x. For simplicity reasons, we use Com to denote the commitment to an element which may be in $\mathbb{G}_1$, $\mathbb{G}_2$ or $\mathbb{Z}_p$. Furthermore, when randomnesses used in a commitment are not essential, we simply write $\mathrm{Com}(X, \_)$.

4) Non-Interactive Zero-Knowledge Proofs: To prove their possession of secret values satisfying given statements without revealing them, both clients and service providers use the proof system proposed by Groth and Sahai [17]. This system allows users to compute Non-Interactive Zero-Knowledge proofs of knowledge (NIZK) on four kinds of statements, including pairing product equations

$$\left(\prod_{i=1}^{n} e(A_i, Y_i)\right)\left(\prod_{j=1}^{m} e(X_j, B_j)\right)\left(\prod_{i=1}^{n}\prod_{j=1}^{m} e(X_j, Y_i)^{\gamma_{ij}}\right) = R_T,$$

and multi-scalar multiplication equations in $\mathbb{G}_1$

$$\left(\prod_{i=1}^{n} {A_i}^{y_i}\right)\left(\prod_{j=1}^{m} {X_j}^{b_j}\right)\left(\prod_{i=1}^{n}\prod_{j=1}^{m} {X_j}^{\gamma_{ij}\, y_i}\right) = R_1,$$

where capital letters denote group elements and lower-case letters denote scalars, *i.e.* elements of $\mathbb{Z}_p$. In these equations, A_i, B_j, b_j, γ_{ij}, R_T, and R_1 are public elements, while X_j, Y_i, and y_i are secret values.

The proofs are based on SXDH commitments to the secret values. From the randomnesses used in the commitments, a prover computes specific group elements (the proof). The verifier can check the validity of the equation using the proof and the committed secrets, without any access to the secrets. A prover can moreover combine multiple statements to prove that secret values simultaneously satisfy different equations. According to the notations introduced by Camenish *et al.* [10], we denote a proof of knowledge of secret values $(x_1, \ldots, x_k)$ following equations $(E_1, \ldots, E_\ell)$ by $\mathrm{NIZK}\{x_1, \ldots, x_k : E_1 \wedge \cdots \wedge E_\ell\}$.

5) Anonymous Proxy Signatures: Throughout their interaction, both parties must sign messages. However, verifying a signature requires to have access to the verification key of the signer, which is incompatible with the privacy properties since verification keys identify signers. An anonymous proxy signature scheme [16] allows receivers to check the validity of signed messages without having access to the identity of the signer. The only obtained information is that the signer received a delegation from an identified authority. To achieve such a property, a signer uses NIZKs to hide all elements related to their identity in the signature. An efficient anonymous proxy signature scheme can be obtained by combining the structure-preserving signature scheme proposed by Abe *et al.* [1] and the proof system proposed by Groth and Sahai [17].

Let $(\mathsf{vk}_\mathcal{U}, \mathsf{sk}_\mathcal{U})$ be the verification and signing key of user $\mathcal{U}$. Then, the signature of message m is $\sigma = \mathrm{Sign}(m, \mathsf{sk}_\mathcal{U})$. The verification consists in the computation of $\mathrm{Verify}(\sigma, m, \mathsf{vk}_\mathcal{U})$: the result is True iff signature σ is valid for the pair $(m, \mathsf{vk}_\mathcal{U})$. We now consider a signing key $\mathsf{sk}_\mathcal{C}$ and the corresponding verification key $\mathsf{vk}_\mathcal{C}$ of a certification authority $\mathcal{C}$. A certificate for user $\mathcal{U}$ is $\mathsf{cert}_\mathcal{U} = \mathrm{Sign}(\mathsf{vk}_\mathcal{U}, \mathsf{sk}_\mathcal{C})$. Such a certificate is valid if and only if $\mathrm{Verify}(\mathsf{cert}_\mathcal{U}, \mathsf{vk}_\mathcal{U}, \mathsf{vk}_\mathcal{C}) = \mathsf{True}$.

Using the structure-preserving property of the signature scheme, an anonymous proxy signature of message m, from an anonymous signer $\mathcal{U}$ certified by the authority $\mathcal{C}$, contains the SXDH commitments to $\mathsf{cert}_\mathcal{U}$, $\mathsf{vk}_\mathcal{U}$ and σ, and a Groth-Sahai proof,

$$\mathrm{NIZK}\big\{\, \mathsf{cert}_\mathcal{U}, \mathsf{vk}_\mathcal{U}, \sigma \;:\; \big(\mathrm{Verify}(\mathsf{cert}_\mathcal{U}, \mathsf{vk}_\mathcal{U}, \mathsf{vk}_\mathcal{C}) = \mathsf{True}\big) \wedge \big(\mathrm{Verify}(\sigma, m, \mathsf{vk}_\mathcal{U}) = \mathsf{True}\big)\big\}.$$

In our context, both clients and service providers compute anonymous proxy signatures to authenticate messages without disclosing their identity. More precisely, they first send commitments to their public key $\mathsf{vk}_\mathcal{U}$ and their certificate $\mathsf{cert}_\mathcal{U}$ together with the proof of registration $\Pi_{\mathsf{cert}_\mathcal{U}}$ given by

$$\Pi_{\mathsf{cert}_\mathcal{U}} = \mathrm{NIZK}\big\{\, \mathsf{cert}_\mathcal{U}, \mathsf{vk}_\mathcal{U} \;:\; \big(\mathrm{Verify}(\mathsf{cert}_\mathcal{U}, \mathsf{vk}_\mathcal{U}, \mathsf{vk}_\mathcal{C}) = \mathsf{True}\big)\big\}.$$

For each message m to be signed, both clients and providers compute a signature σ, and send a commitment to this signature, and a proof of validity Π_σ, that is

$$\Pi_\sigma = \mathrm{NIZK}\big\{\, \mathsf{vk}_\mathcal{U}, \sigma \;:\; \big(\mathrm{Verify}(\sigma, m, \mathsf{vk}_\mathcal{U}) = \mathsf{True}\big)\big\}.$$

The use of the same commitment to $\mathsf{vk}_\mathcal{U}$ is of primary interest here, as the verifier has to check only one proof of registration. Moreover, this commitment establishes a link between the anonymous proxy signatures sent during an interaction: the signer is the same. For convenience reasons, we denote by $\mathrm{APSign}(m, \mathsf{sk}_\mathcal{U})$ the proof Π_σ with the commitment to the signature of m.

6) Verifiable Secret Sharing: Verifiable secret sharing [15] is a threshold cryptographic scheme allowing to split a secret into n different shares. As in a secret sharing scheme, the secret can be recomputed from a predefined number t of shares (with $t \leqslant n$), while giving no information to whoever knows strictly less than t shares. The extra-property consists in the proven consistency of distributed shares: the prover guarantees that the expected secret was shared, and that any set of t shares allows the computation of this unique secret. We now present a verifiable secret sharing scheme for group elements, using both SXDH commitments and NIZKs.

Let $X \in \mathbb{G}_1$ be the secret of the prover, on which this prover has certificate cert_X such that $\mathrm{Verify}(\mathsf{cert}_X, X, \mathsf{vk}_\mathcal{C}) = \mathsf{True}$.

To share X, the prover chooses a $t-1$ degree polynomial $Q : z \in \mathbb{Z}_p \mapsto (X \cdot \prod_{j=1}^{t-1}(A_j)^{z^j}) \in \mathbb{G}_1$, in which the (A_j) are randomly chosen in $\mathbb{G}_1$. The shares are $(i, Q_i = Q(i))_{1 \leqslant i \leqslant n}$. Lagrange interpolation with t distinct shares (i_1, Q_{i_1}), ..., (i_t, Q_{i_t}) allows the reconstruction of secret $X = Q(0)$, as follows

$$X = \mathrm{Interp}\left((i_j, Q_{i_j})_{1 \leqslant j \leqslant t}\right) = \prod_{j=1}^{t} Q_{i_j}{}^{\left(\prod_{k \neq j}(i_k/(i_k - i_j))\right)}.$$

To prove the consistency of the shares, the prover shows that the same polynomial Q is used for the computation of all shares. Thus, the prover commits to the secret X and to the coefficients (A_j), that is, $C_X = \mathrm{Com}(X, \_)$ and $C_{A_j} = \mathrm{Com}(A_j, \_)$ for $1 \leqslant j < t$. The prover then computes NIZKs assessing both the consistency of the shares, *i.e.* a proof Π_{Q_i} that $Q_i = Q(i)$, and the correctness of the secret, using a proof of correctness Π_X including a commitment to the certificate (see details in Section VI). Afterwards, the prover sends i, Q_i, C_X, (C_{A_j}), and Π_{Q_i} to each share carrier, and C_X, (C_{A_j}), C_{cert_X} and Π_X to the verifier. Each share carrier verifies Π_{Q_i} and sends a confirmation to the verifier, that is i, C_X, and (C_{A_j}). If the commitments received from the share carriers and from the prover are the same, then the consistency of the shares is proven. In the meantime, the verifier checks the validity of Π_X.

The verifiable secret sharing is useful iff *(i)* the verifier will eventually receive enough valid signatures, *(ii)* the malicious share carriers cannot recover the secret, and *(iii)* the honest share carriers can recover the secret. By noting b the number of malicious share carriers and ℓ the number of responses that the verifier is waiting for, these conditions translate to $b < t$, $\ell \leqslant n - b$, and $t \leqslant \ell - b$. An optimal choice for t is

$$t = \lceil n/3 \rceil \quad \text{and} \quad \ell = 2t - 1,$$

which tolerates up to $b = t - 1$ malicious share carriers.

Remark 1: Such a sharing can be made *offline*, that is without any interaction with the share carriers before the reconstruction of the secret. In this case, the prover encrypts the shares with the keys of the share carriers to prevent the verifier from reconstructing the secret. To guarantee the correctness of the shares, the prover computes NIZK proofs of their encryption and of their reconstruction. To prevent the verifier from asking the decryption of arbitrary encrypted shares, the prover also signs each share, and computes a NIZK proving the validity of each signature, while masking the share, the signature, and the verification key.

VI. Reputation Protocol

A. System Setup

Throughout the reputation protocol, users need the cryptographic keys and identifiers presented in Table I. Specifically, the central authority $\mathcal{C}$ uses a structure-preserving signature key pair $(\mathsf{vk}_{\mathcal{C}}, \mathsf{sk}_{\mathcal{C}})$ to generate certificates on users' credentials. To enter the system, users register to this authority, which may require a computational or monetary cost [9] to mitigate Sybil attacks. Note that the central authority is required only for the registration of users, and possibly for the choice of accredited signers (see Section V-A2).

Clients have a structure-preserving signature key pair, consisting of a verification key $\mathsf{vk}_{\mathrm{Cl}}$ and a signing key $\mathsf{sk}_{\mathrm{Cl}}$. When clients enter the system, they register to the central authority $\mathcal{C}$ to get a random identifier $\mathsf{id}_{\mathrm{Cl}} \in \mathbb{Z}_p$, and a certificate $\mathsf{cert}_{\mathrm{Cl}}$ on $\mathsf{id}_{\mathrm{Cl}}$ and $\mathsf{vk}_{\mathrm{Cl}}$. Similarly, service providers have a structure-preserving signature key pair $(\mathsf{vk}_{\mathrm{SP}}, \mathsf{sk}_{\mathrm{SP}})$, and register to $\mathcal{C}$ to obtain a random identifier $\mathsf{Id}_{\mathrm{SP}} \in \mathbb{G}_1$, and a certificate $\mathsf{cert}_{\mathrm{SP}}$ on $\mathsf{Id}_{\mathrm{SP}}$ and $\mathsf{vk}_{\mathrm{SP}}$.

Accredited signers have a structure-preserving signature key pair $(\mathsf{vk}_{\mathrm{AS}}, \mathsf{sk}_{\mathrm{AS}})$ and a classical certificate $\mathsf{cert}_{\mathrm{AS}}$ on $\mathsf{vk}_{\mathrm{AS}}$. They use these keys to sign the reputation score of service providers. We denote by σ_i the signature of the i-th accredited signer on the reputation score $\mathsf{rep}_{\mathrm{SP}}$ of the provider, for current round rnd (see Section V-A):

$$\sigma_i = \mathrm{Sign}\left(\langle \mathsf{vk}_{\mathrm{SP}}, \mathsf{rep}_{\mathrm{SP}}, \mathsf{rnd} \rangle, \mathsf{sk}_{\mathrm{AS}_i}\right).$$

Share carriers possess two key pairs, namely a classical encryption key pair $(\mathsf{ek}_{\mathrm{SC}}, \mathsf{dk}_{\mathrm{SC}})$, and a classical signature key pair $(\mathsf{sk}_{\mathrm{SC}}, \mathsf{vk}_{\mathrm{SC}})$, each of them used to protect their communications (encryption of received messages, and signature of sent messages). They also have a certificate $\mathsf{cert}_{\mathrm{SC}}$ on $\mathsf{ek}_{\mathrm{SC}}$ and $\mathsf{vk}_{\mathrm{SC}}$, issued by the central authority $\mathcal{C}$.

Both clients and providers compute by themselves their own pseudonyms. They renew them at each interaction. Pseudonyms are SXDH commitments to their verification keys. Similarly, both clients and service providers compute commitments $C_{\mathsf{id}_{\mathrm{Cl}}}$ and $C_{\mathsf{Id}_{\mathrm{SP}}}$ to their identifiers $\mathsf{id}_{\mathrm{Cl}}$ and $\mathsf{Id}_{\mathrm{SP}}$, leading to

$$\begin{cases} \mathsf{nym}_{\mathrm{Cl}} = \mathrm{Com}(\mathsf{vk}_{\mathrm{Cl}}, \_) \\ C_{\mathsf{id}_{\mathrm{Cl}}} = \mathrm{Com}(\mathsf{id}_{\mathrm{Cl}}, \_), \end{cases} \quad \begin{cases} \mathsf{nym}_{\mathrm{SP}} = \mathrm{Com}\left(\mathsf{vk}_{\mathrm{SP}}, (r_{\mathrm{SP}}, r'_{\mathrm{SP}})\right) \\ C_{\mathsf{Id}_{\mathrm{SP}}} = \mathrm{Com}(\mathsf{Id}_{\mathrm{SP}}, (r_{\mathsf{Id}_{\mathrm{SP}}}, r'_{\mathsf{Id}_{\mathrm{SP}}})). \end{cases}$$

Clients compute commitments $C_{\mathsf{cert}_{\mathrm{Cl}}}$ to their certificate, and NIZK proofs of their validity $\Pi_{\mathsf{cert}_{\mathrm{Cl}}}$ (see Section V-B5). Similarly, service providers compute commitments $C_{\mathsf{cert}_{\mathrm{SP}}}$ and proofs $\Pi_{\mathsf{cert}_{\mathrm{Cl}}}$. Finally, service providers compute a pre-invariant $\mathsf{pre\_inv}$ from $\mathsf{Id}_{\mathrm{SP}}$ and a randomly chosen scalar $r_{\mathsf{pre\_inv}}$: $\mathsf{pre\_inv} = \mathrm{Pre\_inv}(\mathsf{Id}_{\mathrm{SP}}, r_{\mathsf{pre\_inv}})$ (see Section V-B2).

B. Proof of the Reputation Score

When a client wishes to interact with a service provider, he sends a pseudonym $\mathsf{nym}_{\mathrm{Cl}}$ and a proof of its validity $C_{\mathsf{id}_{\mathrm{Cl}}}$, $C_{\mathsf{cert}_{\mathrm{Cl}}}$, and $\Pi_{\mathsf{cert}_{\mathrm{Cl}}}$ to the provider (see Figure 1). Once the provider has verified this proof, she sends back her pseudonym, reputation, pre-invariant and respective proofs of validity. That is, she sends $\mathsf{nym}_{\mathrm{SP}}$, $C_{\mathsf{Id}_{\mathrm{SP}}}$, $C_{\mathsf{cert}_{\mathrm{SP}}}$, $\Pi_{\mathsf{cert}_{\mathrm{SP}}}$, $\mathsf{rep}_{\mathrm{SP}}$, a proof of reputation Π_{rep}, $\mathsf{pre\_inv}$, and a proof $\Pi_{\mathsf{pre\_inv}}$ of its computation. These proofs are defined by

$$\Pi_{\mathsf{rep}} = \mathrm{NIZK}\Big\{ \mathsf{vk}_{\mathrm{SP}}, \sigma_{i_1}, \ldots, \sigma_{i_{t_{\mathrm{AS}}}} : \wedge_{j=1}^{t_{\mathrm{AS}}} \left(\mathrm{Verify}(\sigma_{i_j}, \langle \mathsf{vk}_{\mathrm{SP}}, \mathsf{rep}_{\mathrm{SP}}, \mathsf{rnd} \rangle, \mathsf{vk}_{\mathrm{AS}_{i_j}})\right) \Big\},$$

$$\Pi_{\mathsf{pre\_inv}} = \mathrm{NIZK}\big\{ \mathsf{Id}_{\mathrm{SP}}, r_{\mathsf{pre\_inv}} : \mathsf{pre\_inv} = \mathrm{Pre\_inv}(\mathsf{Id}_{\mathrm{SP}}, r_{\mathsf{pre\_inv}}) \big\}.$$

Table I
USERS' CREDENTIALS (BOLD KEYS ARE SECRET)

	Client		Provider		SC	AS
	Element	Commitment	Element	Commitment		
Signature key	$\mathbf{sk}_{\mathrm{Cl}}$		$\mathbf{sk}_{\mathrm{SP}}$		$\mathbf{sk}_{\mathrm{SC}}$	$\mathbf{sk}_{\mathrm{AS}}$
Verification key	$\mathsf{vk}_{\mathrm{Cl}}$	$\mathsf{nym}_{\mathrm{Cl}}$	$\mathsf{vk}_{\mathrm{SP}}$	$\mathsf{nym}_{\mathrm{SP}}$	$\mathsf{vk}_{\mathrm{SC}}$	$\mathsf{vk}_{\mathrm{AS}}$
Decryption key					$\mathbf{dk}_{\mathrm{SC}}$	
Encryption key					$\mathsf{ek}_{\mathrm{SC}}$	
Invariant	$\mathsf{id}_{\mathrm{Cl}}$	$C_{\mathsf{id}_{\mathrm{Cl}}}$	$\mathsf{Id}_{\mathrm{SP}}$	$C_{\mathsf{Id}_{\mathrm{SP}}}$		
Certificate	$\mathsf{cert}_{\mathrm{Cl}}$	$C_{\mathsf{cert}_{\mathrm{Cl}}}$	$\mathsf{cert}_{\mathrm{SP}}$	$C_{\mathsf{cert}_{\mathrm{SP}}}$	$\mathsf{cert}_{\mathrm{SC}}$	$\mathsf{cert}_{\mathrm{AS}}$
(verified with)	$(\mathsf{vk}_{\mathrm{Cl}}, \mathsf{id}_{\mathrm{Cl}})$	$(\Pi_{\mathsf{cert}_{\mathrm{Cl}}}, \mathsf{nym}_{\mathrm{Cl}}, C_{\mathsf{id}_{\mathrm{Cl}}})$	$(\mathsf{vk}_{\mathrm{SP}}, \mathsf{Id}_{\mathrm{SP}})$	$(\Pi_{\mathsf{cert}_{\mathrm{SP}}}, \mathsf{nym}_{\mathrm{SP}}, C_{\mathsf{Id}_{\mathrm{SP}}})$		
Reputation			$\mathsf{rep}_{\mathrm{SP}}$			
Cert. of reputation			$\{\sigma_1, \ldots, \sigma_{t_{\mathrm{AS}}}\}$	$\Pi_{\mathsf{rep}_{\mathrm{SP}}}$		
(verified with)			$(\mathsf{vk}_{\mathrm{SP}}, \mathsf{rep}_{\mathrm{SP}})$	$(\mathsf{nym}_{\mathrm{SP}}, \mathsf{rep}_{\mathrm{SP}})$		

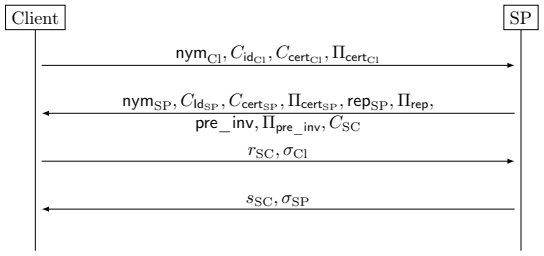

Figure 1. Proof of the reputation score

In the meantime, the provider chooses a nonce s_{SC} and commits to it by sending $C_{\mathrm{SC}} = H(00 \| s_{\mathrm{SC}})$.[1] If the reputation of the provider suits the client, and if all the proofs are valid, the client computes the masked invariant $\mathsf{masked\_inv} = \mathrm{Mask}(\mathsf{pre\_inv}, \mathsf{id}_{\mathrm{Cl}}, r_{\mathsf{masked\_inv}})$, chooses a nonce r_{SC}, signs a hash of $(C_{\mathrm{SC}}, r_{\mathrm{SC}}, \mathsf{nym}_{\mathrm{SP}})$, and sends r_{SC} and the signature σ_{Cl} to the provider with

$$\sigma_{\mathrm{Cl}} = \mathrm{APSign}\left(H(C_{\mathrm{SC}}, r_{\mathrm{SC}}, \mathsf{nym}_{\mathrm{SP}}), \mathsf{sk}_{\mathrm{Cl}}\right).$$

If σ_{Cl} is valid, the provider signs a hash of $(s_{\mathrm{SC}}, r_{\mathrm{SC}}, \mathsf{nym}_{\mathrm{Cl}})$, and sends s_{SC} and the signature σ_{SP} to the client with

$$\sigma_{\mathrm{SP}} = \mathrm{APSign}\left(H(s_{\mathrm{SC}}, r_{\mathrm{SC}}, \mathsf{nym}_{\mathrm{Cl}}), \mathsf{sk}_{\mathrm{SP}}\right).$$

Note that the signatures guarantee that the client agreed to conduct a transaction with provider $\mathsf{nym}_{\mathrm{SP}}$, who uses the randomness hidden in C_{SC}, and that the provider agreed to conduct a transaction with client $\mathsf{nym}_{\mathrm{Cl}}$, who uses randomness r_{SC}. Once the client and the provider have exchanged their nonces, they choose the share carriers as described in Section V-A1, using $(s_{\mathrm{SC}} \| r_{\mathrm{SC}} \| \mathsf{nym}_{\mathrm{Cl}} \| \mathsf{nym}_{\mathrm{SP}})$ as a seed. In the remainder, this element serves as an identifier of the transaction, and we note it $\mathsf{id}_{\mathsf{trans}}$.

C. Sharing Ingredients of the Report

Once both the client and the service provider have agreed to engage in the transaction, they rely on the verifiable secret sharing scheme described in Section V-B6 to guarantee the undeniability properties. The service provider shares her identifier $\mathsf{Id}_{\mathrm{SP}}$, that is, she chooses a polynomial Q of degree $t_{\mathrm{SC}} - 1$, with coefficients $\mathsf{Id}_{\mathrm{SP}}, A_1, \ldots, A_{t_{\mathrm{SC}}-1}$, where the A_j are randomly chosen in $\mathbb{G}_1$. The shares are the $\big(i, Q_i = Q(i)\big)$ for $1 \leq i \leq n_{\mathrm{SC}}$. To prove the sharing, the provider computes commitments C_{A_j} to the A_j, and NIZK proofs Π_{Q_i} defined by

$$\Pi_{Q_i} = \mathrm{NIZK}\left\{ \mathsf{Id}_{\mathrm{SP}}, (A_j)_j : Q_i = \mathsf{Id}_{\mathrm{SP}} \prod_{j=1}^{t_{\mathrm{SC}}-1} {A_j}^{(i^j)} \right\},$$

for $1 \leq i \leq n_{\mathrm{SC}}$. Note that $\mathsf{nym}_{\mathrm{SP}}$, $C_{\mathsf{Id}_{\mathrm{SP}}}$, $C_{\mathsf{cert}_{\mathrm{SP}}}$ and $\Pi_{\mathsf{cert}_{\mathrm{SP}}}$ have already proven the correctness of the secret. Finally, the provider sends the (C_{A_j}) to the client, and encrypts and sends $\mathsf{id}_{\mathsf{trans}}$, (i, Q_i), $C_{\mathsf{Id}_{\mathrm{SP}}}$, $(C_{A_j})_{1 \leq j < t_{\mathrm{SC}}}$, and Π_{Q_i} to the i-th share-carrier. If the received proof is valid, the share carriers send a confirmation to the client, that is $\mathsf{id}_{\mathsf{trans}}$, i, $C_{\mathsf{Id}_{\mathrm{SP}}}$, and (C_{A_j}), together with a signature. If these commitments are the same as the one received from the provider, the client accepts this confirmation. Once the client has received $\ell = 2\lceil n_{\mathrm{SC}}/3 \rceil - 1$ valid shares, he accepts the sharing (as the validity of the shares guarantees the undeniability properties). Figure 2 describes this exchange.

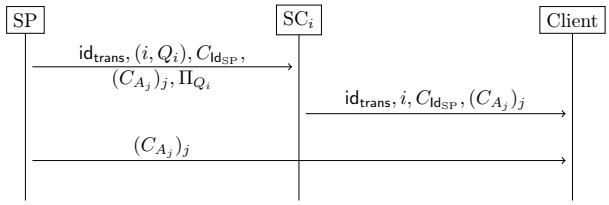

Figure 2. Secretly Sharing the Provider's Identifier

In the meantime, the client shares his secret, that is the masked invariant $\mathsf{masked\_inv}$. Since $\mathsf{masked\_inv}$ consists of two elements, he must double the sharing. That is, the client chooses two polynomial $R_{\mathbf{1}}$, $R_{\mathbf{2}}$ of degree $t_{\mathrm{SC}} - 1$ with coefficients $\mathsf{masked\_inv}_{\mathbf{k}}, B_{1,\mathbf{k}}, \ldots, B_{t_{\mathrm{SC}}-1,\mathbf{k}}$ for $\mathbf{k} \in \{\mathbf{1}, \mathbf{2}\}$, and the shares are $\big(i, R_i = \big(R_{\mathbf{1}}(i), R_{\mathbf{2}}(i)\big)\big)$ for $1 \leq i \leq n_{\mathrm{SC}}$. To prove the sharing, the client computes commitments $C_{\mathsf{masked\_inv}}$ and $C_{B_{j,\mathbf{k}}}$ to $\mathsf{masked\_inv}$ and to the $B_{j,\mathbf{k}}$, and

[1] This concatenation guarantees that s_{SC} and r_{SC} are chosen independently.

NIZK proofs Π_{R_i} defined by

$$\Pi_{R_i} = \mathrm{NIZK}\left\{ \mathsf{masked\_inv}, (B_{j,\mathbf{k}}) \ : \ R_{i,\mathbf{k}} = \mathsf{masked\_inv}_{\mathbf{k}} \cdot \prod_{j=1}^{t_{\mathrm{SC}}-1} {B_{j,\mathbf{k}}}^{(i^j)}, \mathbf{k} \in \{\mathbf{1}, \mathbf{2}\}\right\},$$

for $1 \leq i \leq n_{\mathrm{SC}}$. To prove the correctness of the secret, the client also computes a proof $\Pi_{C_{\mathsf{masked\_inv}}}$ defined by

$$\Pi_{C_{\mathsf{masked\_inv}}} = \mathrm{NIZK}\left\{ \mathsf{id}_{\mathrm{Cl}}, \mathsf{masked\_inv}, r_{\mathsf{masked\_inv}} \ : \ \mathsf{masked\_inv} = \mathrm{Mask}(\mathsf{pre\_inv}, \mathsf{id}_{\mathrm{Cl}}, r_{\mathsf{masked\_inv}})\right\}.$$

Thus, the client sends $C_{\mathsf{masked\_inv}}$, $(C_{B_{j,\mathbf{k}}})$, and $\Pi_{C_{\mathsf{masked\_inv}}}$ to the provider, and encrypts and sends $\mathsf{id}_{\mathsf{trans}}$, (i, R_i), $C_{\mathsf{masked\_inv}}$, $(C_{B_{j,\mathbf{k}}})$, and Π_{R_i} to the i-th share carrier. As previously, the i-th share carrier sends a confirmation consisting of $\mathsf{id}_{\mathsf{trans}}$, i, $C_{\mathsf{masked\_inv}}$, $(C_{B_{j,\mathbf{k}}})$, and a signature to the provider if the share is valid. The provider accepts such a confirmation if the commitments are identical to the ones she received, and accepts the sharing as soon as she has received ℓ valid confirmations. Figure 3 describes this exchange. Once both sharings have been accepted, both parties can conduct their transaction.

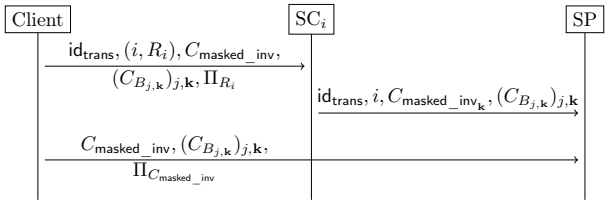

Figure 3. Secretly Sharing the Masked Invariant

D. Issuing Reports

Once the transaction is over, the client can issue a rating and the provider can obtain a proof of transaction. Scenario A describes their interactions.

a) Scenario A – Standard case.: The client chooses a rating ρ and computes both a signature $\sigma_{\rho,\mathrm{Cl}}$ to prevent any modification on ρ, and a proof $\Pi_{\mathsf{masked\_inv}}$ of $\mathsf{masked\_inv}$ computation. We have

$$\sigma_{\rho,\mathrm{Cl}} = \mathrm{APSign}\left(\langle \mathsf{id}_{\mathsf{trans}}, \rho\rangle, \mathsf{sk}_{\mathrm{Cl}}\right),$$
$$\Pi_{\mathsf{masked\_inv}} = \mathrm{NIZK}\left\{r_{\mathsf{masked\_inv}}, \mathsf{id}_{\mathrm{Cl}} : \mathsf{masked\_inv} = \mathrm{Mask}(\mathsf{pre\_inv}, \mathsf{id}_{\mathrm{Cl}}, r_{\mathsf{masked\_inv}})\right\}.$$

Since $\mathsf{masked\_inv}$ no longer needs to be hidden, this proof is simpler than $\Pi_{C_{\mathsf{masked\_inv}}}$. As described in Figure 4, the client sends message m_1 to the provider, with $m_1 = (\mathsf{id}_{\mathsf{trans}}, \rho, \mathsf{masked\_inv}, \Pi_{\mathsf{masked\_inv}}, \sigma_{\rho,\mathrm{Cl}})$. If both the proof and signature are valid, the provider computes the invariant $\mathsf{inv} = \mathrm{Unmask}(\mathsf{masked\_inv}, r_{\mathsf{pre\_inv}})$, and a signature $\sigma_{\rho,\mathrm{SP}}$ with

$$\sigma_{\rho,\mathrm{SP}} = \mathrm{Sign}\left(\langle \mathsf{id}_{\mathsf{trans}}, \sigma_{\rho,\mathrm{Cl}}\rangle, \mathsf{sk}_{\mathrm{SP}}\right).$$

Note that $\sigma_{\rho,\mathrm{SP}}$ guarantees that the client chose his rating before knowing the identifier of the provider, which guarantees his objectivity. The provider then reveals her identifier and the randomnesses used in $\mathsf{nym}_{\mathrm{SP}}$, $C_{\mathsf{Id}_{\mathrm{SP}}}$, and $\mathsf{pre\_inv}$, that is $\Pi_{\mathrm{SP}} = (r_{\mathrm{SP}}, r'_{\mathrm{SP}}, r_{\mathsf{Id}_{\mathrm{SP}}}, r'_{\mathsf{Id}_{\mathrm{SP}}}, r_{\mathsf{pre\_inv}})$. The provider sends message m_2 to the client, with $m_2 = (\mathsf{Id}_{\mathrm{SP}}, \mathsf{vk}_{\mathrm{SP}}, \mathsf{cert}_{\mathrm{SP}}, \mathsf{inv}, \Pi_{\mathrm{SP}}, \sigma_{\rho,\mathrm{SP}})$. The client verifies that

$$\begin{cases} \mathsf{nym}_{\mathrm{SP}} = \mathrm{Com}\left(\mathsf{vk}_{\mathrm{SP}}, (r_{\mathrm{SP}}, r'_{\mathrm{SP}})\right) \\ C_{\mathsf{Id}_{\mathrm{SP}}} = \mathrm{Com}(\mathsf{Id}_{\mathrm{SP}}, (r_{\mathsf{Id}_{\mathrm{SP}}}, r'_{\mathsf{Id}_{\mathrm{SP}}})) \\ \mathsf{pre\_inv} = \mathrm{Pre\_inv}(\mathsf{Id}_{\mathrm{SP}}, r_{\mathsf{pre\_inv}} \\ \mathsf{inv} = \mathrm{Unmask}(\mathsf{masked\_inv}, r_{\mathsf{pre\_inv}}). \end{cases}$$

Finally, both the client and the provider are able to issue the report by sending the elements given in the first column of Table II to the share carriers (where the first four lines represent the proof of transaction and the last one the rating together with the signatures of both parties). If all the signatures and proofs are valid, the report itself is considered valid by the share carriers. This scenario completes successfully if both parties are honest. If the client does not send message m_1 (resp. the provider does not send message m_2) then scenario B (resp. scenario C) is run.

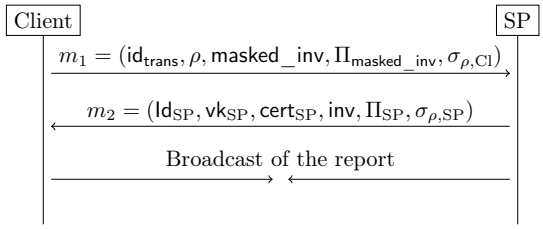

Figure 4. Interactions between the client and the provider to jointly issue their report

b) Scenario B – Dishonest client/honest provider: If the provider does not receive message m_1 from the client, she queries the share carriers for their share. On their turn, they query the client to get his rating and, in absence of his answer, send their shares and associated proofs to the provider. The provider is able to reconstruct the masked invariant as $\mathsf{masked\_inv} = \mathrm{Interp}\left((i_j, R_{i_j})_{1 \leqslant j \leqslant t_{\mathrm{SC}}}\right)$ from t valid received shares. From that point, the service provider computes $\mathsf{inv} = \mathrm{Unmask}(\mathsf{masked\_inv}, r_{\mathsf{pre\_inv}})$, and issues the report, which only contains the proof of transaction (*i.e.*, the elements in the second column of Table II). Figure 5 describes this interaction.

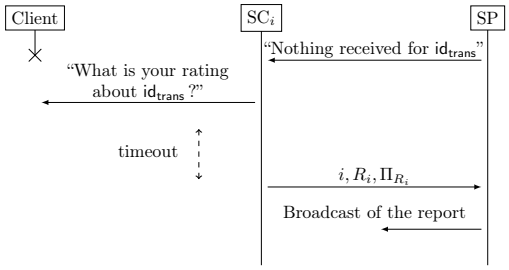

Figure 5. Interactions between the provider and the share carriers to issue the report

Table II
COMPONENTS OF THE REPORT IN THE THREE SCENARII

	Scenario A (honest users)	Scenario B (dishonest client)	Scenario C (dishonest provider)
Service provider	$\mathsf{Id}_{\mathrm{SP}}$, $\mathsf{vk}_{\mathrm{SP}}$, $\mathsf{cert}_{\mathrm{SP}}$, $\mathsf{nym}_{\mathrm{SP}}$, $C_{\mathsf{Id}_{\mathrm{SP}}}$, Π_{SP}	$\mathsf{Id}_{\mathrm{SP}}$, $\mathsf{vk}_{\mathrm{SP}}$, $\mathsf{cert}_{\mathrm{SP}}$, $\mathsf{nym}_{\mathrm{SP}}$, $C_{\mathsf{Id}_{\mathrm{SP}}}$, Π_{SP}	$C_{\mathsf{Id}_{\mathrm{SP}}}$, $\mathsf{nym}_{\mathrm{SP}}$, $\mathsf{PCert}_{\mathrm{SP}}$, $\mathsf{Id}_{\mathrm{SP}}$, $(C_{A_j})_j$, $\{i_j, Q_{i_j}, \Pi_{Q_{i_j}}\}_j$
Client	$C_{\mathsf{id}_{\mathrm{Cl}}}$, $\mathsf{nym}_{\mathrm{Cl}}$, $\mathsf{PCert}_{\mathrm{Cl}}$	$C_{\mathsf{id}_{\mathrm{Cl}}}$, $\mathsf{nym}_{\mathrm{Cl}}$, $\mathsf{PCert}_{\mathrm{Cl}}$	$C_{\mathsf{id}_{\mathrm{Cl}}}$, $\mathsf{nym}_{\mathrm{Cl}}$, $\mathsf{PCert}_{\mathrm{Cl}}$
Transaction identifier	$\mathsf{id}_{\mathsf{trans}}, \sigma_{\mathrm{SP}}, \sigma_{\mathrm{Cl}}$	$\mathsf{id}_{\mathsf{trans}}, \sigma_{\mathrm{SP}}, \sigma_{\mathrm{Cl}}$	$\mathsf{id}_{\mathsf{trans}}, \sigma_{\mathrm{SP}}, \sigma_{\mathrm{Cl}}$
Invariant	$\mathsf{pre\_inv}$, $\mathsf{masked\_inv}$, inv, $r_{\mathsf{pre\_inv}}$, $\Pi_{\mathsf{masked\_inv}}$	$\mathsf{pre\_inv}$, $\mathsf{masked\_inv}$, inv, $r_{\mathsf{pre\_inv}}$, $(C_{B_{j,\mathbf{k}}})$, $\{i_j, R_{i_j}, \Pi_{R_{i_j}}\}_j$, $C_{\mathsf{masked\_inv}}$, $\Pi_{C_{\mathsf{masked\_inv}}}$	inv, Π_{inv}
Rating	ρ, $\sigma_{\rho,\mathrm{Cl}}$, $\sigma_{\rho,\mathrm{SP}}$		ρ, $\{\sigma_{\rho,\mathrm{SC}_{i_j}}\}_j$

c) Scenario C – Dishonest provider/honest client: If the client does not receive message m_2 from the provider, he informs the share carriers by sending them the masked invariant and his rating together with the associated proofs and signatures as shown in Figure 6. If the proofs and signatures are valid, the share carriers forward them to the provider to give her the opportunity to reveal $\mathsf{Id}_{\mathrm{SP}}$ and the invariant. In absence of any response, the share carriers send their shares to the client. Note that they also sign the rating of the client to validate the fact that the client has chosen his rating before knowing the provider's identity. We have

$$\sigma_{\rho,\mathrm{SC}_i} = \mathrm{Sign}\left(\langle \mathsf{id}_{\mathsf{trans}}, \sigma_{\rho,\mathrm{Cl}} \rangle, \mathsf{sk}_{\mathrm{SC}_i}\right).$$

Once the client has received t valid shares, he computes $\mathsf{Id}_{\mathrm{SP}} = \mathrm{Interp}\left((i_j, Q_{i_j})_{1 \leqslant j \leqslant t_{\mathrm{SC}}}\right)$, $\mathsf{inv} = \mathrm{Inv}(\mathsf{Id}_{\mathrm{SP}}, \mathsf{id}_{\mathrm{Cl}})$, and a proof Π_{inv} of its computation, with

$$\Pi_{\mathsf{inv}} = \mathrm{NIZK}\left\{ \mathsf{id}_{\mathrm{Cl}} : \mathsf{inv} = {\mathsf{Id}_{\mathrm{SP}}}^{\mathsf{id}_{\mathrm{Cl}}} \right\}.$$

Finally, the client issues the report by sending the elements presented on the third column of Table II to the share carriers.

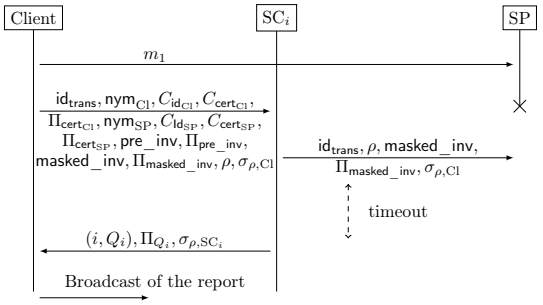

Figure 6. Interactions between the client and the share carriers to issue the report

Note that if neither the client nor the service provider issue the report, then the transaction is not taken into account in the reputation score of the service provider.

Remark 2: One may argue that both parties can collude by jointly issuing reports as soon as the preparation is over, that is, without having actually conducted a transaction. Similarly, clients may choose arbitrary ratings regardless of the behavior of providers. The only way to prevent such behaviors would be to have an omniscient and trusted third party capable of telling whether the transaction really occurred, or whether the effort exerted by the provider during her transaction was bad or good enough to deserve such a rating, which is clearly impossible.

E. Computation of the Reputation Scores

At the end of round rnd, each share carrier gathers all the reports received since round $\mathsf{rnd}-1$, and sends them to the accredited signers. This allows the accredited signers to update the reputation scores of all the service providers concerned by valid reports. Once accredited signers have checked the validity of a report, they only keep the identifier of the provider, identifier of the transaction, the invariant inv, and the rating of the client, if any, and sign them. Note that if two (or more) reports have the same identifier of transaction and invariant, they keep a single one. Beyond handling negative ratings, the accredited signers know the rounds during which reports have been cast. Thus, as motivated in Section II, any reputation score function can be used, *e.g.* to lower the influence of old ratings [21] or to limit the impact of ballot-stuffing attacks [33]. In addition, the accredited signers approximate the reputation score of service providers to extend their anonymity set. Once the accredited signers have computed the reputation score of a provider, they sign it along with the round, rnd, and send it to the service provider:

$$\sigma_i = \mathrm{Sign}\left(\langle \mathsf{vk}_{\mathrm{SP}}, \mathsf{rep}_{\mathrm{SP}}, \mathsf{rnd} \rangle, \mathsf{sk}_{\mathrm{AS}_i}\right).$$

Service providers can then use these signatures to prove their reputation to their clients during round $\mathsf{rnd}+1$.

VII. PERFORMANCE EVALUATION

We now evaluate our privacy-preserving reputation mechanism both in theoretical and practical ways. The former evaluation is achieved through an analysis of the performance of each building block, while the latter relies on its implementation on a platform made of heterogeneous computing nodes. The number of share carriers n_{SC} and the number of accredited signers n_{AS} are respectively equal to $n_{\mathrm{SC}} = 28$ and $n_{\mathrm{AS}} = 1$. This setting is sufficient to prevent the collusions of $\lceil n_{\mathrm{SC}}/3 \rceil - 1 = 9$ share carriers with probability 2^{-20} in

Table III
SIZE OF EXCHANGED MESSAGES FOR $n_{\mathrm{SC}} = 28$ AND $n_{\mathrm{AS}} = 1$, IN KIBIBYTES

Phase	Cl $\leftrightarrow$ SP	Cl $\leftrightarrow$ SC$_i$	SP $\leftrightarrow$ SC$_i$	report
Proof of Reputation	22	0	0	—
Sharing	3.28	2.69	2.34	—
Scenario A	2.94	0	0	12.06
Scenario B	0	0	0.75	19.63
Scenario C	0	17.94	0	20.75

a system comprising 10^8 service providers, including 5×10^6 malicious ones, as explained in Appendix A.

A. Theoretical Study

The correctness of our mechanism relies on the verification of NIZK proofs, which requires the computation of many pairings. To decrease the number of these operations, we adopt the technique proposed by Blazy *et al.* [8] which consists in verifying NIZKs by batches. We also ensure efficient pairing computations by relying on prime-order elliptic curves [5]. We consider elliptic curves in a subclass of the Barreto-Naehrig family. Thus, elements of $\mathbb{Z}_p$ and $\mathbb{G}_1$ (resp. $\mathbb{G}_2$) can be represented by 32 B (resp. 64 B). We use the computation costs given by Aranha *et al.* [5]. Namely, the four cores of a 3.0 GHz AMD Phenom II X4 940 processor – a top-level processor of 2010 – can compute 8 pairings in a millisecond, 16 exponentiations in $\mathbb{G}_2$, or 48 in $\mathbb{G}_1$. In the following, we study two metrics, namely (a) the size of messages exchanged between each entity, and (b) the time necessary for each entity to perform his computation. We now present and comment the main results obtained with these settings. Table III gives the size of messages (in KiB) exchanged between the different parties involved in the reputation mechanism, namely, between the client and the provider, the client and one share carrier, and the provider and one share carrier before the transaction takes place. Finally, it gives the size of the report sent to the accredited signer once the transaction is over.

These results are both satisfactory and reassuring. The largest messages correspond to the proof of reputation, which comprises the mutual authentication of the service provider and the client, and the proof by the provider of his reputation score. Nevertheless, this exchange requires only around 20 KiB. This is impressive compared to the mechanism proposed by Bethencourt *et al.* [7], where the proof of reputation requires 500 KiB per received rating. Table III also shows that share carriers only need 3 KiB when a transaction goes well, and less than 10 KiB in the worst case situation. This clearly shows that the design of a distributed trusted third party requires very little resources. The same comment applies for the accredited signers. The size of the report, that comprises all the proofs, requires no more than 20 KiB in the worst case. It is important to note that the only message that scales (linearly) as a function of the number of the accredited signers is the proof of reputation. Thus, even for larger sets of accredited signers, which typically do not grow to more than 20 entities, the communication cost remains acceptable. These results are very reassuring because they show that, from a theoretical point of view, privacy-preserving reputation mechanisms are entirely viable. The next section will show that this holds in practice!

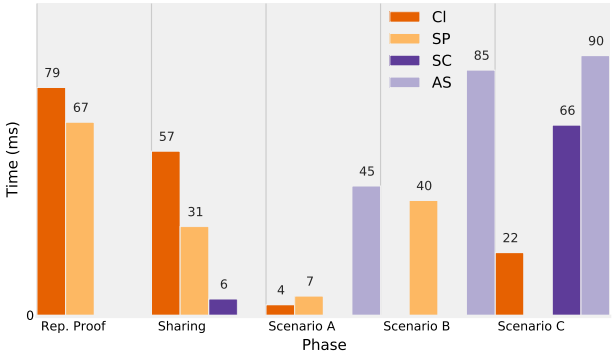

Figure 7. Theoretical computation times (ms)

Figure 7 details the computation cost (in ms) of each phase of the reputation mechanism at each of the involved entities. Several remarks are in order. The main one is that computation times are very low. Indeed, each user needs no more than 200 ms for all their computations. In particular, each share carrier needs no more than 6 ms when both the client and the provider are honest. Even in the worst case, they need only 75 ms to perform their computations. Finally, the verification of a report requires between 45 ms and 90 ms. This clearly shows that participating to one of the two distributed trusted third parties computing entities costs little. Actually, the largest costs are due to scenarii B or C. We can minimize those costs by penalizing malicious users, *e.g.* by preventing them from interacting for a given period of time.

B. Implementing the Reputation Mechanism

We have implemented our reputation mechanism in Python 2.7 with the Charm framework [2]. This framework facilitates the implementation of complex cryptographic primitives, such as Groth-Sahai's NIZK proof system [17], and the combination of multiple primitives, *e.g.*, to build anonymous proxy signatures [16]. Furthermore, Charm provides the means to benchmark applications, both by giving their running time and by counting each elementary cryptographic operation. We also use Twisted, an event-driven networking engine, to handle communications between the different parties. Experiments have been conducted on heterogeneous entities, namely, a virtual machine running on a Dell Latitude E6430 laptop with a 2.60 GHz Core i7-3720 QM processor, and cheap Raspberry Pi model B machines with the Raspbian operating system.

Figure 8 presents the results of the conducted experiments. It shows the mean and standard deviation of the computation times of each user for every step of the interaction, namely, the proof of reputation, the sharing, and the issuing of the report in every scenario. Note that the "SC" columns correspond to the computation times of one share carrier running on the virtual

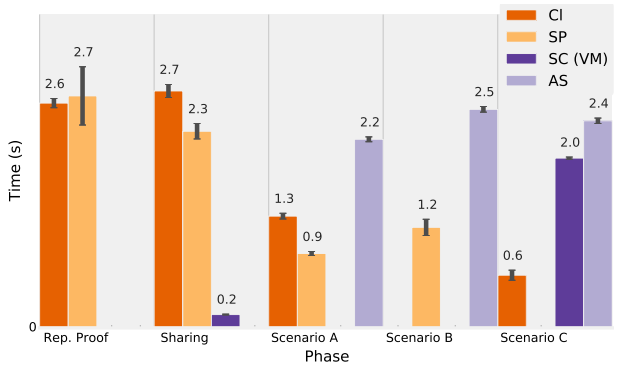

Figure 8. Practical computation times (s)

machine, and that the "AS" columns relate to the verification of one report by an accredited signer.

Clearly, the computation times are higher than the one obtained in theory, which can be easily explained. First, Aranha *et al.* carefully select a Barreto-Naehrig curve, and optimize the computation of pairings using Assembly and C code [5] on this specific curve. In our case, we rely on the MNT-159 curve proposed by Charm, which is a Python framework wrapping around Lynn's `pbc` library.[2] Furthermore, the theoretical number of operations per second assumes that they are all ran in parallel, which is not the case in our experiments. Finally, all the users except one share carrier were run on a single virtual machine. This does not slow down the phases where users run computations sequentially, *e.g.* the proof of reputation or the construction of the report in Scenario A, but it does slow down the concurrent ones, *e.g.* the sharing of the secrets.

Even with those limitations, we observe that our mechanism allows clients to interact with providers, and to run all the preparation in no more than 5 s. Issuing the report may take longer, but the most important point is that clients can rapidly verify the reputation of a provider and get involved in the transaction. Similarly, the pre-transaction and the post-transaction phases respectively require no more than 5 s and 1 s which clearly allows the provider to interact with many clients simultaneously. Note that share carriers can even be run on cheap Raspberry Pi machines. In that case, sharing the secrets requires no more than 4.7 s, while issuing the rating in presence of malicious clients needs no more than 59 s. Such cheap machines increase the waiting time of both clients and providers, but this delay remains acceptable (less than 15 s), compared for example to the time required to buy items on any e-commerce web sites. Finally, running clients on Raspberry Pi requires about 75 s for conducting the reputation proof and 115 s for the sharing. That is, clients need about 3 min before being able to conduct a transaction, which is clearly reasonable to engage in (possibly) financial transactions.

[2]http://crypto.stanford.edu/pbc/

VIII. Conclusion

In this article, we have presented a practical distributed reputation mechanism addressing two main issues of reputation mechanisms: preserving all users' privacy *and* computing reputation scores based on both positive and negative ratings. This has been achieved by combining distributed algorithms with cryptographic schemes. Furthermore, our proposition is independent of the reputation model, that is, our system can integrate any reputation model [21], preferably one using both positive and negative ratings.

As future works, we plan to study more deeply the off-line version of the secret sharing, in particular to improve the report verification when the service provider does not want to collaborate. We also plan to study whether the presence of a unique trusted entity is a necessary condition to handle both non monotonic reputation scores and the permanent anonymity of providers.

References

[1] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo. Structure-preserving signatures and commitments to group elements. In T. Rabin, editor, *Advances in Cryptology—CRYPTO*, pages 209–236, Santa Barbara, California, USA, Aug. 2010. Springer Berlin Heidelberg.

[2] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin. Charm: a framework for rapidly prototyping cryptosystems. *Journal of Cryptographic Engineering*, 3(2):111–128, 2013.

[3] E. Anceaume and A. Ravoaja. Incentive-based robust reputation mechanism for P2P services. In A. A. Shvartsman, editor, *International Conference on Principles of Distributed Systems (OPODIS)*, pages 305–319, Bordeaux, France, Dec. 2006. Springer Berlin Heidelberg.

[4] E. Androulaki, S. G. Choi, S. M. Bellovin, and T. Malkin. Reputation systems for anonymous networks. In N. Borisov and I. Goldberg, editors, *Privacy Enhancing Technologies*, pages 202–218, Leuven, Belgium, July 2008. Springer Berlin Heidelberg.

[5] D. F. Aranha, K. Karabina, P. Longa, C. H. Gebotys, and J. López. Faster explicit formulas for computing pairings over ordinary curves. In K. G. Paterson, editor, *Eurocrypt*, pages 48–68, Talinn, Estonia, May 2011. Springer Berlin Heidelberg.

[6] R. F. Baumeister, E. Bratslavsky, C. Finkenauer, and K. D. Vohs. Bad is stronger than good. *Review of general psychology*, 5(4):323–370, 2001.

[7] J. Bethencourt, E. Shi, and D. Song. Signatures of reputation. In R. Sion, editor, *Financial Cryptography and Data Security*, pages 400–407, Tenerife, Canary Islands, Jan. 2010. Springer Berlin Heidelberg.

[8] O. Blazy, G. Fuchsbauer, M. Izabachène, A. Jambert, H. Sibert, and D. Vergnaud. Batch groth-sahai. In *Applied Cryptography and Network Security (ACNS)*, Beijing, China, June 2010. Springer Berlin Heidelberg.

[9] N. Borisov. Computational puzzles as sybil defenses. In A. Montresor, A. Wierzbicki, and N. Shahmehri, editors, *IEEE International Conference on Peer-to-Peer Computing*, pages 171–176, Cambridge, United Kingdom, Oct. 2006. IEEE Computer Society.

[10] J. Camenisch and M. Stadler. Efficient group signature schemes for large groups (extended abstract). In B. S. Kaliski Jr., editor, *Advances in Cryptology—CRYPTO*, pages 410–424, Santa Barbara, California, USA, Aug. 1997. Springer Berlin Heidelberg.

[11] S. Clauß, S. Schiffner, and F. Kerschbaum. *k*-anonymous reputation. In K. Chen, Q. Xie, W. Qiu, N. Li, and W.-G. Tzeng, editors, *ACM Symposium on Information, Computer and Communications Security (ASIACCS)*, pages 359–368. ACM, May 2013.

[12] C. Dellarocas. Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior. In J. K. MacKie Mason and D. Tygar, editors, *ACM Conference on Electronic Commerce*, pages 150–157, Minneapolis, Minnesota, USA, Oct. 2000. ACM.

[13] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In M. Blaze, editor, *USENIX Security Symposium*, pages 303–320, San Diego, California, USA, Aug. 2004. USENIX.

[14] European Parliament and Council of the European Union. Directive 95/46/EC. *Official Journal of the European Communities*, L281:31–50, Nov. 1995.

[15] P. Feldman. A practical scheme for non-interactive verifiable secret sharing. In *Foundations of Computer Science*, pages 427–437, Los Angeles, California, USA, Oct. 1987. IEEE Computer Society.

[16] G. Fuchsbauer and D. Pointcheval. Anonymous proxy signatures. In R. Ostrovsky, R. D. Prisco, and I. Visconti, editors, *Security and Cryptography for Networks*, pages 201–217, Amalfi, Italy, Sept. 2008. Springer Berlin Heidelberg.

[17] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In N. P. Smart, editor, *Eurocrypt*, pages 415–432, Istanbul, Turkey, Apr. 2008. Springer Berlin Heidelberg.

[18] E. Hamilton, M. Kriens, H. Karapandžić, K. Yaici, M. Main, and S. Schniffer. Report on trust and reputation models. Technical report, European Network and Information Security Agency (ENISA), Dec. 2011.

[19] O. Hasan, L. Brunie, and E. Bertino. Preserving privacy of feedback providers in decentralized reputation systems. *Computers & Security*, 31(7):816–826, 2012.

[20] O. Hasan, L. Brunie, E. Bertino, and N. Shang. A decentralized privacy preserving reputation protocol for the malicious adversarial model. *IEEE Transactions on Information Forensics and Security*, 8(6):949–962, 2013.

[21] A. Jøsang and R. Ismail. The beta reputation system. In *Bled Electronic Commerce Conference*, Bled, Slovenia, June 2002.

[22] A. Jøsang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, Mar. 2007.

[23] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The EigenTrust algorithm for reputation management in P2P networks. In *International World Wide Web Conference*, pages 640–651, Budapest, Hungary, May 2003. ACM.

[24] F. Kerschbaum. A verifiable, centralized, coercion-free reputation system. In E. Al-Shaer and S. Paraboschi, editors, *Workshop on Privacy in the Electronic Society*, pages 61–70, Chicago, Illinois, USA, Nov. 2009. ACM.

[25] T. Mahler and T. Olsen. Reputation systems and data protection law. In P. Cunningham and M. Cunningham, editors, *eAdoption and the Knowledge Economy: Issues, Applications, Case Studies*, pages 180–187. IOS Press, Oct. 2004.

[26] S. Marti and H. Garcia-Molina. Taxonomy of trust: Categorizing P2P reputation systems. *Computer Networks*, 50(4):472–484, Mar. 2006.

[27] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *IEEE Symposium on Security and Privacy*, pages 111–125, Oakland, California, USA, May 2008. IEEE Computer Society.

[28] National Institute of Standards and Technology. Secure hash standard (SHS), Mar. 2012.

[29] E. Pavlov, J. S. Rosenschein, and Z. Topol. Supporting privacy in decentralized additive reputation systems. In C. D. Jensen, S. Poslad, and T. Dimitrakos, editors, *Trust Management*, pages 108–119. Springer Berlin Heidelberg, Apr. 2004.

[30] A. Pfitzmann and M. Hansen. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf, Aug. 2010. v0.34.

[31] P. Resnick and R. Zeckhauser. Trust among strangers in Internet transactions: Empirical analysis of eBay's reputation system. *The Economics of the Internet and E-Commerce*, 11:127–157, Nov. 2002.

[32] S. Steinbrecher. Enhancing multilateral security in and by reputation systems. In V. Matyás, S. Fischer-Hübner, D. Cvrcek, and P. Svenda, editors, *The Future of Identity in the Information Society*, pages 135–150, Brno, Czech Republich, Sept. 2008. Springer Berlin Heidelberg.

[33] A. Whitby, A. Jøsang, and J. Indulska. Filtering out unfair ratings in bayesian reputation systems. In *Proceedings of the 7th International Workshop on Trust in Agent Societies*, New York, New York, USA, Nov. 2004.

[34] B. Yu and M. P. Singh. Distributed reputation management for electronic commerce. *Computational Intelligence*, 18(4):535–549, Nov. 2002.

APPENDIX

A. Number of Share Carriers

The share carriers are users that guarantee the undeniability of reports. They are jointly chosen by clients and service providers for a single interaction. As explained in Section V-A1, they are randomly chosen among all service providers. Hence, the choice of the share carriers is equivalent to a draw without replacement, which can be modeled by the hypergeometric distribution. Let N be the number of potential share carriers and m the number of malicious users. Then, the probability of having chosen less than a third of malicious share carriers corresponds to $1 - \mathrm{cdf}_{N,m,n_{\mathrm{SC}}}(t_{\mathrm{SC}} - 1)$, where $\mathrm{cdf}_{N,m,n_{\mathrm{SC}}}$ is the cumulative distribution function of the hypergeometric distribution with parameters (N, m, n_{SC}).[3] Therefore, if we wish to achieve a given maximal probability of collusion $p_{\max}$, we must choose

$$n_{\mathrm{SC}} = \min\left\{n \;\middle|\; 1 - \mathrm{cdf}_{N,m,n}\left(\left\lceil\frac{n}{3}\right\rceil - 1\right) < p_{\max}\right\}.$$

Figure 9 represents n_{SC} for N varying between 100 and 10^{15}, for fixed percentages m of malicious users, and for $p_{\max} = 2^{-20}$.

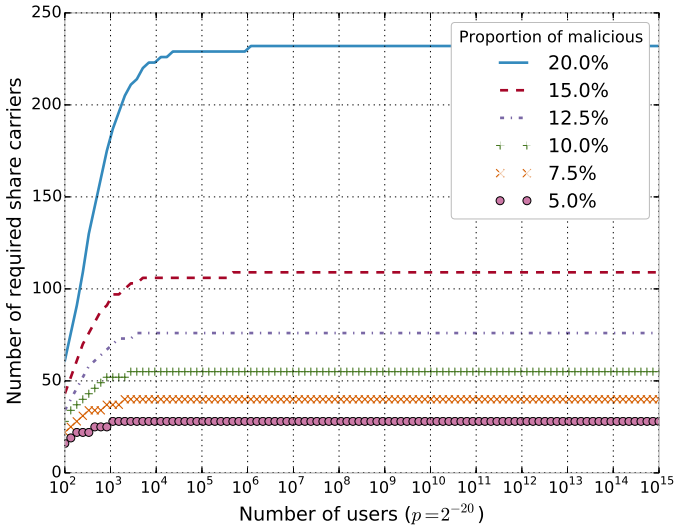

Figure 9. Dependance of n_{SC} in N and m ($p_{\max} = 2^{-20}$)

As we can see, the number of required share carriers does not grow after the number of users has reached 1,000,000. This means that whatever the size of the system, the number of required share carriers does not grow and our mechanism scales. Figure 10 represents n_{SC} for $p_{\max}$ varying between 2^{-10} and 2^{-70}, and $N = 10^8$. As we can see, the number of required share carriers is logarithmic in the maximal probability of collusion targeted. For instance, with $n_{\mathrm{SC}} = 100$, we can either either prevent collusions representing 15% of total users from obtaining a third of malicious share carriers with probability 2^{-20}, or prevent collusions representing 10% of total users with probability 2^{-35}. Since the share carriers

[3] We showed in Section V-B6 that $t_{\mathrm{SC}} = \lceil n_{\mathrm{SC}}/3 \rceil$.

are renewed for each transaction, it is acceptable that a report in a million is either falsified or denied. We can thus take $p_{\max} = 2^{-20}$.

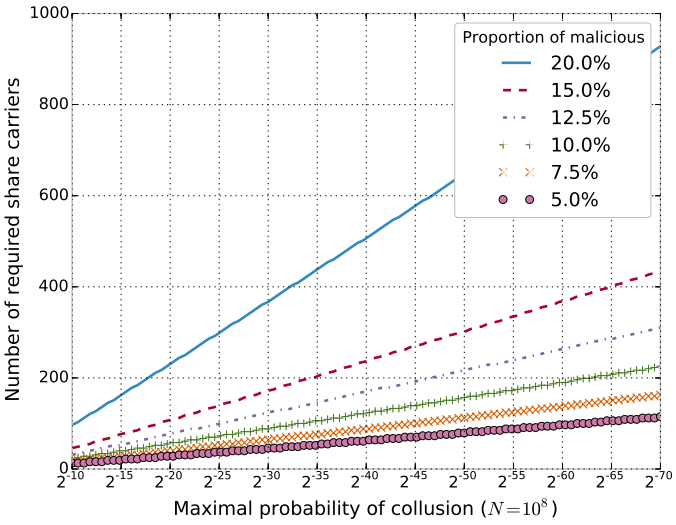

Figure 10. Dependance of n_{SC} in $p_{\max}$ and m ($N = 10^8$)

Finally, Figure 11 presents the number of required share carriers for $N = 10^8$ and $p_{\max} = 2^{-20}$, for varying m. As we can see, to keep a reasonable number of share carriers, our mechanism can tolerate up to 5% colluding users, which requires 28 share carriers. That is, the probability that enough share carriers are chosen among the 10^7 colluding ones is lower than 2^{-20}.

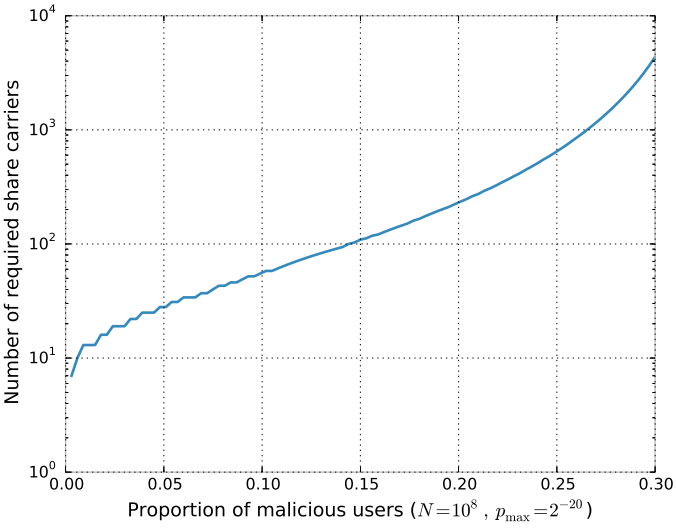

Figure 11. Dependance of n_{SC} in m ($p_{\max} = 2^{-20}$ and $N = 10^8$)

B. Cryptographic Proofs

In this appendix, we present proofs that our reputation mechanism guarantees the properties presented in Section III. Note that these proofs are not the most formal. In particular, we do not present the setups of the underlying schemes. To prevent interferences, each cryptographic tool should be setup independently. Furthermore, each cryptographic tool should have different setups for each role. That is, there should be one setup for NIZKs constructed by clients and another one for those constructed by providers, and similarly for every cryptographic tool.

Before presenting the formal definitions and proofs of our properties, we must discuss the model of the adversary, that we call $\mathcal{A}$. First, we consider $\mathcal{A}$ to be *probabilistic polynomial-time Turing machine*. We want the adversary to be able to: *(i)* make two users interact; *(ii)* corrupt users, since the adversary might be a collusion of users; and *(iii)* know which client was involved in a transaction. Hence, we consider three oracles that $\mathcal{A}$ might query during the games:

(i) $\mathcal{O}_{\mathsf{inter}}$ that, given a client and a provider, makes them interact;
(ii) $\mathcal{O}_{\mathsf{corr}}$ that, given a user, returns all this user's credentials;
(iii) $\mathcal{O}_{\mathsf{inv}}$ that, given an invariant, returns the client involved.

Note that the winning conditions of each game will depend on the oracles queried by the adversary. For instance, the adversary must not corrupt a user to compromise this user's privacy.

C. Privacy of Service Providers

Privacy of service providers – Property 1 – is guaranteed if the following experiment returns 1 with probability $\frac{1}{2}+\mathrm{neg}(\lambda)$, where neg is a negligible function:

1) $(\mathsf{state}) \leftarrow \mathrm{Setup}(1^\lambda)$
2) $(\mathsf{state}, \mathrm{SP}_0, \mathrm{SP}_1) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{inter}}(\cdot),\mathcal{O}_{\mathsf{corr}}(\cdot),\mathcal{O}_{\mathsf{inv}}(\cdot)}(\mathsf{state})$
3) $b \leftarrow \{0,1\}$
4) $b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{inter}}(\cdot),\mathcal{O}_{\mathsf{corr}}(\cdot),\mathcal{O}_{\mathsf{inv}}(\cdot)}(\mathsf{state}, \mathrm{SP}_b, \mathrm{SP}_{\bar{b}})$
5) Return 1 if $b' = b$; 0 otherwise.

In Step 2 of this game, $\mathcal{A}$ can make any two users interact. $\mathcal{A}$ can also corrupt any users, but strictly less than a majority of n_{AS} accredited signers. Service providers SP_0 and SP_1 are service providers of equivalent reputation that were not corrupted by $\mathcal{A}$. In Step 4, $\mathcal{A}$ cannot corrupt SP_0, SP_1, SP_b, $\mathrm{SP}_{\bar{b}}$ or more than t_{SC} share carriers involved in one of SP_b's or $\mathrm{SP}_{\bar{b}}$'s interactions. $\mathcal{A}$ also cannot make SP_b or $\mathrm{SP}_{\bar{b}}$ interact with a client and go beyond the report; thus, there is no invariant computed by SP_b or $\mathrm{SP}_{\bar{b}}$, and SP_b and $\mathrm{SP}_{\bar{b}}$ have not revealed their identity to any client.

Proof: The only way for $\mathcal{A}$ to obtain information regarding b is to have clients interact with SP_b. However, $\mathcal{A}$ does not see neither the invariant nor the identity of SP_b. Furthermore, making SP_b interact simultaneously with many different clients does not give more information to $\mathcal{A}$ than just one interaction: all the elements shown by SP_b are randomized. Therefore, we consider a single interaction between SP_b and a client chosen by $\mathcal{A}$, and we assume that the last phase of the transaction has not started yet.

During this interaction, $\mathcal{A}$ sees $\mathsf{nym}_{\mathrm{SP}_b}$, $C_{\mathsf{Id}_{\mathrm{SP}_b}}$, $\mathsf{rep}_{\mathrm{SP}_b}$, $\mathsf{pre\_inv}_b$, $\{\mathsf{vk}_{\mathrm{AS}_{i_k}}\}_{1\leqslant k\leqslant t_{\mathrm{AS}}}$, $C_{\mathsf{cert}_{\mathrm{SP}_b}}$, $\Pi_{\mathsf{cert}_{\mathrm{SP}_b}}$, Π_{rep_b}, $\Pi_{\mathsf{pre\_inv},b}$, $s_{\mathrm{SC},b}$, $\sigma_{\mathrm{SP},b}$, $\{C_{A_{j,b}}\}_{1\leqslant j\leqslant t_{\mathrm{SC}}-1}$, $\{Q_{i,b}, \Pi_{c,i,b}\}_{i\in I}$, where I is a set of cardinal lower than t_{SC}. Since $\mathcal{A}$ has

corrupted less than t_{SC} share carriers, the $Q_{i,b}$ give no information about $\mathsf{Id}_{\mathrm{SP}_b}$ and can be considered as random elements.

Among those elements, $s_{\mathrm{SC},b}$ and the $Q_{i,b}$ are random elements, hence they give no information about the service provider. Using the hiding property of commitments and NIZKs, new games can be defined, in which all commitments and NIZKs – hence anonymous proxy signatures – are replaced by random elements. The indistinguishability of these games with the original ones is related to the security assumption of the commitment and NIZK schemes. The only element that might leak information is $\mathsf{pre\_inv}_b$, that is $({G_1}^r, \mathsf{Id}_{\mathrm{SP}_b}{Y_1}^r)$, where r is randomly chosen by SP_b. From the DDH assumption in $\mathbb{G}_1$ on the tuple $(G_1, {G_1}^r, Y_1 = {G_1}^{\alpha_1})$, it follows that ${Y_1}^r$ is indistinguishable from a random element $R \in \mathbb{G}_1$. By taking $R' = {\mathsf{Id}_{\mathrm{SP}_1}}^{-1} \cdot \mathsf{Id}_{\mathrm{SP}_0} \cdot R$, we have $\mathsf{Id}_{\mathrm{SP}_0} \cdot R = \mathsf{Id}_{\mathrm{SP}_1} \cdot R'$, where R' is also indistinguishable from Y_2^r. Thus, $\mathsf{Id}_{\mathrm{SP}_0} \cdot {Y_1}^r$ is indistinguishable from $\mathsf{Id}_{\mathrm{SP}_1} \cdot {Y_1}^r$, and the advantage of the adversary in this game is lower than two times the DDH advantage.

The same reasoning can be applied to $\mathrm{SP}_{\bar{b}}$. ∎

D. Privacy of Clients

Privacy of clients (Property 2) relies on two parts. First, during a transaction, the provider does not know whom they are interacting with. Secondly, the transactions between a client and different service providers are unlinkable. The following two experiments capture these notions. This property is guaranteed if both return 1 with probability $\frac{1}{2} + \mathrm{neg}(\lambda)$.

1) $(\mathsf{state}) \leftarrow \mathrm{Setup}(1^\lambda)$
2) $(\mathsf{state}, \mathrm{Cl}_0, \mathrm{Cl}_1, \{\mathrm{SP}_i\}_{1 \leqslant i \leqslant n}) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{inter}}(\cdot), \mathcal{O}_{\mathsf{corr}}(\cdot), \mathcal{O}_{\mathsf{inv}}(\cdot)}(\mathsf{state})$
3) $b \leftarrow \{0, 1\}$
4) $b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{inter}}(\cdot), \mathcal{O}_{\mathsf{corr}}(\cdot), \mathcal{O}_{\mathsf{inv}}(\cdot)}(\mathsf{state}, \mathrm{Cl}_b, \mathrm{Cl}_{\bar{b}}, \{\mathrm{SP}_i\}_{1 \leqslant i \leqslant n})$
5) Return 1 if $b' = b$; 0 otherwise.

In Step 2 of this experiment, Cl_0 and Cl_1 are clients who have not been corrupted by $\mathcal{A}$. In Step 4, $\mathcal{A}$ cannot corrupt Cl_0, Cl_1, Cl_b, or $\mathrm{Cl}_{\bar{b}}$. Furthermore, the set of service providers having interacted (and gone beyond the transaction) with Cl_0 or Cl_1 is disjoint from the set of providers having interacted (and gone beyond the transaction) with Cl_b or $\mathrm{Cl}_{\bar{b}}$. That is, no service provider has interacted with (Cl_0 or Cl_1) *and* (Cl_b or $\mathrm{Cl}_{\bar{b}}$).

Proof: The proof of this property contains two parts. Firstly, can the adversary obtain any information about b without having the clients go beyond the transaction? Secondly, can the adversary link the clients of different providers?

Similarly to the previous proof, if the clients do not go beyond the transaction, $\mathcal{A}$ sees only commitments, NIZK proofs, anonymous proxy signatures, and randomnesses. As explained previously, these elements give no information about b or $\bar{b}$.

After the transaction, the only different elements are the masked invariants, and the invariants:

$$\mathsf{masked\_inv} = ({G_1}^s {Y_1}^{\mathsf{id}_{\mathrm{Cl}}}, {\mathsf{Id}_{\mathrm{SP}}}^{\mathsf{id}_{\mathrm{Cl}}} {G_1}^{rs} {Y_1}^{r\,\mathsf{id}_{\mathrm{Cl}}})$$
$$\mathsf{inv}_0 = {\mathsf{Id}_{\mathrm{SP}}}^{\mathsf{id}_{\mathrm{Cl}}}$$

Note that since the provider proves the computation of pre_inv, an honest client will not compute anything more than this masked invariant. Furthermore, we remark that the masked invariant is based on the invariant, and ${G_1}^s {Y_1}^{\mathsf{id}_{\mathrm{Cl}}}$. Since s is chosen randomly by the client, this last element is indistinguishable from a random element, and only the final invariant is useful to $\mathcal{A}$.

Now, the adversary $\mathcal{A}$ only has access to the invariants involving Cl_0, Cl_1, Cl_b, or $\mathrm{Cl}_{\bar{b}}$ with the SP_i, with the restriction that a given service provider cannot produce an invariant with (Cl_0 or Cl_1) and (Cl_b or $\mathrm{Cl}_{\bar{b}}$). Thus, we consider a hybrid sequence composed of k service providers involved only with Cl_0 and Cl_1, and $n-k$ other providers involved only with Cl_b and $\mathrm{Cl}_{\bar{b}}$:

Game G_0: $\big(\mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_b), \mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_{\bar{b}})\big), 1 \leqslant i \leqslant n$

Game G_1: $\big(\mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_0), \mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_1)\big), 1 \leqslant i \leqslant n$

Game $\widetilde{G}_k^1$
$$\begin{cases} \big(\mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_0), \mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_1)\big) & 1 \leqslant i \leqslant k \\ \big(\mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_b), \mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_{\bar{b}})\big) & k < i \leqslant n, \end{cases}$$

Game $\widetilde{G}_k^2$
$$\begin{cases} \big(\mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_0), \mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_1)\big) & 1 \leqslant i < k \\ \big(R \leftarrow \mathbb{G}_1, \mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_{\bar{b}})\big) & i = k \\ \big(\mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_b), \mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_{\bar{b}})\big) & k < i \leqslant n, \end{cases}$$

Game $\widetilde{G}_k^3$
$$\begin{cases} \big(\mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_0), \mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_1)\big) & 1 \leqslant i < k \\ \big(R \leftarrow \mathbb{G}_1, R' \leftarrow \mathbb{G}_1\big) & i = k \\ \big(\mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_b), \mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_{\bar{b}})\big) & k < i \leqslant n, \end{cases}$$

Game $\widetilde{G}_k^4$
$$\begin{cases} \big(\mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_0), \mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_1)\big) & 1 \leqslant i < k \\ \big(\mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_0), R' \leftarrow \mathbb{G}_1\big) & i = k \\ \big(\mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_b), \mathrm{Inv}(\mathrm{SP}_i, \mathrm{Cl}_{\bar{b}})\big) & k < i \leqslant n, \end{cases}$$

for $1 \leqslant k \leqslant n$, where R and R' are taken uniformly at random. Note that G_0 and $\widetilde{G}_0^1$ are the same, as well as G_1 and $\widetilde{G}_n^1$. We prove that for $k \in \{0, \ldots, n\}$, for $i \in \{1, 2, 3\}$, $\widetilde{G}_k^i$ and $\widetilde{G}_k^{i+1}$ are indistinguishable, as well as $\widetilde{G}_k^4$ and $\widetilde{G}_{k+1}^1$. Hence, games G_0 and G_1 are indistinguishable.

First, let us prove that games $\widetilde{G}_k^1$ and $\widetilde{G}_k^2$ are indistinguishable. The only difference between those two two games concerns SP_k: the adversary has access to either $\mathsf{inv}(\mathrm{SP}_k, \mathrm{Cl}_0)$ or to a random element R.

Let us consider a DDH tuple $(A = G^a, B = G^b, Z)$. We fix $\mathsf{Id}_{\mathrm{SP}_k} = B$ and $\mathsf{id}_{\mathrm{Cl}_0} = a$. If the adversary $\mathcal{A}$ is able to distinguish these two games, $\mathcal{A}$ is also able to distinguish G^{ab} from a random element R. That is, the adversary can tell whether Z is G^{ab} or a random element, knowing only G, G^a, and G^b.

Hence, $\widetilde{G}_k^1$ and $\widetilde{G}_k^2$ are indistinguishable for $\mathcal{A}$. With the same reasoning, we can show that all the successive games are indistinguishable for $\mathcal{A}$. Therefore, the advantage of the adversary in this hybrid sequence is lower than $4n$ times the DDH advantage, and games G_0 and G_1 are indistinguishable. ∎

E. Undeniability of Reports

The undeniability of ratings (see Property 3), captures the notion that a service provider cannot prevent a client from rating her, and that the client's rating will be taken into account in the provider reputation score. Property 3 is guaranteed if the following game returns 1 with probability $\mathrm{neg}(\lambda)$:

1) $(\mathsf{state}) \leftarrow \mathrm{Setup}(1^\lambda)$
2) $(\mathrm{Cl}_0) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{inter}}(\cdot),\mathcal{O}_{\mathsf{corr}}(\cdot),\mathcal{O}_{\mathsf{inv}}(\cdot)}(\mathsf{state})$
3) $\mathcal{A}$ makes Cl_0 interact with SP_0, and Cl_0 issues a report rpt_0
4) Return 1 if Cl_0 has not been corrupted by $\mathcal{A}$, and either rpt_0 is not valid or rpt_0 was not issued on SP_0

Proof: A game-based approach can formally prove the undeniability of reports, however we only give hints for such a proof. First, we consider the elements received by the client before the transaction. Under the robustness of the signature scheme, and the binding property of the commitment scheme and NIZK proof system, the adversary is not able to impersonate an uncorrupted service provider. Thus, the adversary necessarily uses the credentials of one of the corrupted providers, and must share this provider's identity with the share carriers. After the transaction, when the adversary completes the protocol, the binding property once more guarantees that the revealed identity is the one used before the transaction, which correspond to one of the corrupted service providers. In the other case, the adversary cannot prevent the client from obtaining the share carriers' shares, and then reconstructing the secret. In both cases, the client can construct a valid report. ∎

Similar experiment and reasoning can be used to prove Property 4.

F. Unforgeability of Reports

An adversary has two options to forge a report. First, by forging signatures from a majority of the accredited signers. By assumption, less than $\frac{n_{\mathrm{AS}}}{2}$ accredited signers are malicious, thus such a case is not possible. Secondly, the adversary may forge a valid report. The unforgeability of reports – Property 5 – is guaranteed if the following experiment returns 1 with probability $\mathrm{neg}(\lambda)$:

1) $(\mathsf{state}) \leftarrow \mathrm{Setup}(1^\lambda)$
2) $\mathsf{rpt}_0 \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{inter}}(\cdot),\mathcal{O}_{\mathsf{corr}}(\cdot),\mathcal{O}_{\mathsf{inv}}(\cdot)}(\mathsf{state})$
3) Return 1 if
   a) rpt_0 is valid
   b) $\mathsf{vk}_{\mathrm{Cl}_0} = \mathrm{Open}(\mathsf{nym}_{\mathrm{Cl},0})$
   c) either Cl_0 or SP_0 has not been corrupted by $\mathcal{A}$
   d) rpt_0 is different from any other report between Cl_0 and SP_0

where rpt_0 is a report issued by $\mathsf{nym}_{\mathrm{Cl},0}$ on provider SP_0.

Proof: The report rpt_0 consists of the following four elements:

- the identity of the service provider, represented by $\mathsf{Id}_{\mathrm{SP}_0}$ and $\mathsf{vk}_{\mathrm{SP}_0}$, and their pseudonym for the transaction, that is $C_{\mathsf{Id}_{\mathrm{SP}_0}}$ and $\mathsf{nym}_{\mathrm{SP}_0}$;
- the pseudonym of the client, $\mathsf{nym}_{\mathrm{Cl}_0}$ and $C_{\mathsf{id}_{\mathrm{Cl}_0}}$;
- the identifier of the transaction, $\mathsf{id}_{\mathsf{trans},0}$;
- the invariant, inv_0

And, if the client is correct, the report also comprises their rating ρ_0. Each of these elements is either signed by both the client and the provider – like $\mathsf{id}_{\mathsf{trans}_0}$ – or proven thanks to proofs that may be non-interactive and zero-knowledge, or not – like inv_0. We show that to change any of these elements, $\mathcal{A}$ must break one of the underlying primitives.

a) Changing the identity of the service provider: Suppose that $\mathcal{A}$ has modified $\mathsf{Id}_{\mathrm{SP}}$, $\mathsf{vk}_{\mathrm{SP}}$ and $\mathsf{cert}_{\mathrm{SP}}$. Thus, either $\mathcal{A}$ has forged Π_{SP}, or modified $\mathsf{nym}_{\mathrm{SP}_0}$ and $C_{\mathsf{Id}_{\mathrm{SP}_0}}$, and computed a correct proof Π_{SP}. The security of NIZKs prevents $\mathcal{A}$ from doing the former. Furthermore, σ_{SP} and σ_{Cl} are two signatures from both the provider and the client on $\mathsf{id}_{\mathsf{trans}_0} = s_{\mathrm{SC}} \| r_{\mathrm{SC}} \| \mathsf{nym}_{\mathrm{Cl}} \| \mathsf{nym}_{\mathrm{SP}}$. The security of the anonymous proxy signatures hence prevents $\mathcal{A}$ from forging such signatures. If the service provider did not participate in the report emission, the client proves the reconstruction of $\mathsf{Id}_{\mathrm{SP}_0}$ thanks to the shares of the share carriers. Each of these shares is signed by the service provider, which prevents $\mathcal{A}$ from forging them. Therefore, $\mathcal{A}$ cannot modify the provider's identity. The same reasoning shows that $\mathcal{A}$ cannot modify the identity of the client or the identifier of the transaction either.

b) Changing the invariant: If the service provider did not participate in the report emission, changing the invariant comes down to modifying $\mathsf{Id}_{\mathrm{SP}_0}$ or $\mathsf{id}_{\mathrm{Cl}_0}$ – which is hard, as shown previously – or forging Π_{inv_0}, which is hard as well. If the service provider did participate, inv_0 depends directly from $\mathsf{masked\_inv}_0$. Modifying inv_0 thus means modifying $\mathsf{masked\_inv}_0$. $\mathsf{masked\_inv}_0$ is computed from $\mathsf{pre\_inv}_0$, which depends directly on $\mathsf{Id}_{\mathrm{SP}_0}$. As shown previously, $\mathsf{Id}_{\mathrm{SP}_0}$ cannot be modified. Thus, to modify $\mathsf{masked\_inv}_0$, $\mathcal{A}$ must forge $\Pi_{\mathsf{masked\_inv}_0}$, which is hard. Therefore, $\mathcal{A}$ cannot modify the invariant.

c) Changing the rating: To modify the rating, $\mathcal{A}$ must either forge signatures σ_{d,Cl_0} and σ_{d,SP_0} if both the client and the provider are correct, or forge t signatures from the share carriers. Since $\mathcal{A}$ cannot corrupt both Cl_0 and SP_0, the security of the anonymous proxy signature scheme prevents them to do so.

Since the adversary must break the NIZK proofs or the anonymous proxy signatures to win this game, their advantage is negligible and the unforgeability of reports is guaranteed. ∎

G. Linkability of Reports

The linkability of reports (Property 6) is guaranteed if the following experiment returns 1 with probability $\mathrm{neg}(\lambda)$:

1) $(\mathsf{state}) \leftarrow \mathrm{Setup}(1^\lambda)$
2) $(\mathsf{rpt}_0, \mathsf{rpt}_1) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{inter}}(\cdot),\mathcal{O}_{\mathsf{corr}}(\cdot),\mathcal{O}_{\mathsf{inv}}(\cdot)}(\mathsf{state})$
3) Return 1 if:
   a) $\mathsf{rpt}_0, \mathsf{rpt}_1$ are valid, and
   b) rpt_0 and rpt_1 concern the same provider, and
   c) $\mathrm{Open}(\mathsf{nym}_{\mathrm{Cl},0}) = \mathrm{Open}(\mathsf{nym}_{\mathrm{Cl},1})$ and $\mathsf{inv}_0 \neq \mathsf{inv}_1$, or

d) $\mathrm{Open}(\mathsf{nym}_{\mathrm{Cl},0}) \neq \mathrm{Open}(\mathsf{nym}_{\mathrm{Cl},1})$ and $\mathsf{inv}_0 = \mathsf{inv}_1$

Proof: We previously proved that reports are unforgeable. Thus, if rpt_0 (resp. rpt_1) is valid, rpt_0 contains $\mathsf{inv}_0 = {\mathsf{Id}_{\mathrm{SP}_0}}^{\mathsf{id}_{\mathrm{Cl}_0}}$ (resp. $\mathsf{inv}_1 = {\mathsf{Id}_{\mathrm{SP}_0}}^{\mathsf{id}_{\mathrm{Cl}_1}}$), where $\mathrm{Cl}_0 = \mathrm{Open}(\mathsf{nym}_{\mathrm{Cl},0})$ (resp. $\mathrm{Cl}_1 = \mathrm{Open}(\mathsf{nym}_{\mathrm{Cl},1})$). Therefore, $\mathsf{inv}_0 = \mathsf{inv}_1$ if and only if $\mathrm{Cl}_0 = \mathrm{Cl}_1$, and reports are linkable. ■

H. Representativeness of Reputation Scores

The representativeness of reputation scores (Property 7), captures the notion that an adversary cannot prove another reputation than their own one. This property is guaranteed if the following game returns 1 with probability $\mathrm{neg}(\lambda)$:

1) $(\mathsf{state}) \leftarrow \mathrm{Setup}(1^\lambda)$
2) $(\mathsf{nym}_{\mathrm{SP},0}, C_{\mathsf{Id}_{\mathrm{SP},0}}, \mathsf{rep}_{\mathrm{SP},0}, C_{\mathsf{cert}_{\mathrm{SP},0}}, \Pi_{\mathsf{cert}_{\mathrm{SP},0}}, \Pi_{\mathsf{rep},0}) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{inter}}(\cdot),\mathcal{O}_{\mathsf{corr}}(\cdot),\mathcal{O}_{\mathsf{inv}}(\cdot)}(\mathsf{state})$
3) Return 1 if:
   a) $\Pi_{\mathsf{cert}_{\mathrm{SP},0}}$ and $\Pi_{\mathsf{rep},0}$ are valid for $\mathsf{nym}_{\mathrm{SP},0}$, $C_{\mathsf{Id}_{\mathrm{SP},0}}$, $C_{\mathsf{cert}_{\mathrm{SP},0}}$ and $\mathsf{rep}_{\mathrm{SP},0}$, and
   b) $\mathrm{SP}'_0 = \mathrm{Open}(\mathsf{nym}_{\mathrm{SP},0})$ and $\mathsf{rep}_{\mathrm{SP}'_0} \neq \mathsf{rep}_{\mathrm{SP},0}$

Proof: As proven previously, reports cannot be forged, and thus the reports signed by the accredited signers are legitimate reports. Hence, honest accredited signers compute and sign a representative reputation score. Since we made the assumption that less than $n_{\mathrm{AS}}/2$ accredited signers are malicious, a service provider cannot receive enough signatures from accredited signers to prove a different reputation score. Furthermore, the security of NIZKs and anonymous proxy signatures prevent $\mathcal{A}$ from forging valid proofs of reputation. Therefore, the representativeness of reputation scores holds. ■