Signatures of Reputation (Extended Abstract)

John Bethencourt[1], Elaine Shi[2], and Dawn Song[1]

[1] UC Berkeley {bethenco,dawnsong}@cs.berkeley.edu
[2] PARC eshi@parc.com

Abstract. Reputation systems have become an increasingly important tool for highlighting quality information and filtering spam within online forums. However, the dependence of a user's reputation on their history of activities seems to preclude any possibility of anonymity. We show that useful reputation information can, in fact, coexist with strong privacy guarantees. We introduce and formalize a novel cryptographic primitive we call *signatures of reputation* which supports monotonic measures of reputation in a completely anonymous setting. In our system, a user can express trust in others by voting for them, collect votes to build up her own reputation, and attach a proof of her reputation to any data she publishes, all while maintaining the unlinkability of her actions.

1 Introduction

In various forms, reputation has become a ubiquitous tool for improving the quality of online discussions. For example, a user may mark a product review on Amazon or a business review on Yelp as "useful", and these ratings allow others to more easily identify the best reviews and reviewers. Most web message boards also include a means of providing feedback to help highlight quality content, an early example being Slashdot's "karma" system.

Unfortunately, in all such systems, a user is linked by their pseudonym to a history of their messages or other activities. In many online communities (e.g., a support group for victims of abuse), users may hope that the use of a pseudonym allows them to remain anonymous. However, recent work has shown that very little prior information about an individual is necessary to match them to their pseudonym [1–3]. Building a truly private forum requires abandoning the notion of persistent identities.

We raise the question of whether it is possible to gain all the utility of existing reputation systems while maintaining the unlinkability and anonymity of individual user actions, thus avoiding the histories of activity which threaten privacy. Such a system would enable a number of intriguing applications. For example, we might imagine an anonymous message board in which every post stands alone – not even associated with a pseudonym. Users would rate posts based on whether they are helpful or accurate, collect reputation from other users' ratings, and annotate or sign new posts with the collected reputation.

Other users could then judge new posts based on the author's reputation while remaining unable to identify the earlier posts from which it was derived. Such a forum would allow effective filtering of spam and highlighting of quality information while providing an unprecedented level of user privacy.

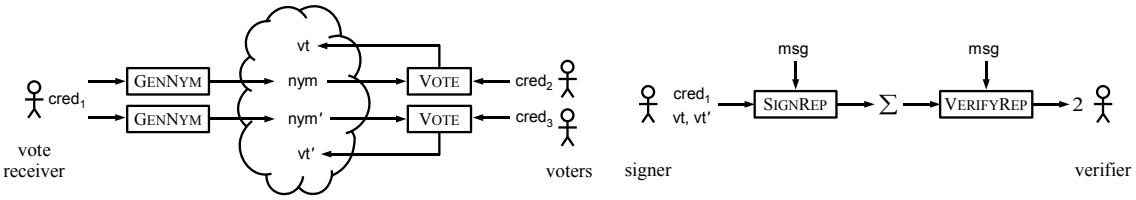
Our approach. To build toward this goal, we propose *signatures of reputation* as a new cryptographic framework enabling the counter-intuitive combination of reputation and anonymity. In a conventional signature scheme, a signature is associated with a public key and convinces the verifier that the signer knows the corresponding private key. Based on the public key, a verifier could then retrieve the reputation of the signer. Through signatures of reputation, we aim to eliminate the middle step of identifying the signer: instead, verification of the signature directly reveals the signer's reputation. With such a tool, a user may apply their reputation to *any* data that they wish to publish online, without risking their privacy. By formally defining this setting, we hope to spur further research into techniques for its realization.

As a first step, we introduce a construction for signatures of reputation that supports *monotonic* aggregation of reputation. That is, we assume that additional feedback cannot decrease a user's reputation. While a user's misbehavior cannot damage reputation they have already accumulated, such a system is sufficient to prevent more casual attackers who, for example, wish to post spam without taking the time to obtain reputation first. Although some existing reputation systems are monotonic (e.g., Google's PageRank algorithm), one would ultimately hope to support non-monotonic reputation as well. We leave this as a primary open problem for future work.

In our construction, the reputation feedback takes the form of cryptographic "votes" that users construct and send to one another, and a user's reputation is simply the number of votes they have collected from distinct users. Each user stores the votes they have collected, and to anonymously sign a message with their reputation, the user constructs a non-interactive zero-knowledge (NIZK) proof of knowledge which demonstrates possession of some number of votes.

The ability of a reputation system to limit the influence of any single user is crucial in enabling applications to control abuse. To this end, our construction ensures that each user can cast at most one valid vote for another user (or up to k for any fixed $k \geq 1$). Enforcing this property is a major technical problem due to the tension with the desired unlinkability properties. In Section 4, we give a high-level overview of the techniques our construction uses to address this and other technical challenges. For details of the construction, proofs, and additional material, we refer the reader to the full version of this paper [4].

Related work. While we are not aware of any work directly comparable to our proposed signatures of reputation, others have explored the conflict between reputation and unlinkability [5–7]. E-cash schemes also attempt to maintain the unlinkability of individual user interactions, and in several cases [8–10] they have been applied for reputation or incentive purposes. The work of Androulaki et al. [10] is particularly close to ours in its aims. However, this and all other e-cash based approaches are incapable of supporting the type of abuse resistance



(a) A user anonymously posts two one-time pseudonyms, each of which receives a vote.

(b) The user may sign a message while proving they have votes from two distinct users.

Fig. 1. One-time pseudonyms, votes, and signatures of reputation.

provided by our scheme because they allow a single user to give multiple coins to another, inflating their reputation. In our scheme, it is possible to prove that a collection of votes came from *distinct* users. This ability to prove distinctness while maintaining the mutual anonymity of both voters and vote receivers is the key technical achievement of our construction.

Anonymous credentials schemes [11–13] may also be considered an effort toward the goal of "trust without identity". There are two key distinctions, however. First, anonymous credentials are concerned with the setting of access control based on trust derived from explicit authorities, whereas this work aims to support trust derived from a very different source: the aggregate opinions of other users. Second, like e-cash based approaches, existing anonymous credential schemes lack a mechanism for proving that votes or credentials come from distinct users while simultaneously hiding the identities of those users.

Finally, our setting superficially resemblances that of e-voting (e.g., [14]), in that it allows the casting of votes while maintaining privacy properties. However, e-voting schemes are designed for an election scenario in which the candidates have no need to receive votes and prove possession of votes anonymously, among other differences, and cannot be used to achieve the properties we require.

2 Defining Signatures of Reputation

We now introduce our formulation of signatures of reputation within the vote counting scenario, then define the algorithms which constitute such a scheme.

Overview. In the system illustrated in Fig. 1, we refer to each user as a *vote receiver*, *voter*, *signer*, or *verifier* depending on their role in the specific algorithm being discussed. To ensure *receiver anonymity*, a vote receiver invokes the GenNym algorithm to compute a "one-time pseudonym" called a nym, which they attach to some content that they publish and wish to receive credit for. A voter can then use the Vote algorithm on a nym to produce a vote which hides their identity, even from the recipient (referred to as *voter anonymity*). The voter posts the vote online where the recipient can later retrieve it. After collecting some votes, a signer runs the SignRep algorithm on a given message to construct a signature of reputation, which must not reveal the signer's identity (*signer anonymity*). We also ensure that a malicious signer cannot inflate its reputation (*reputation soundness*).

To participate in the system, each user must contact a *registration authority* (RA) which generates the user's private credentials, just as the key generating server does within IBE schemes. Although our construction requires trust in the RA for both privacy and reputation soundness, it need only be trusted when registering users and may thereafter go offline. As with typical IBE schemes, it is also possible to reduce the trust necessary in the RA by distributing it amongst multiple parties [15]. Devising a scheme which maintains privacy in the presence of a malicious RA is an interesting problem for future work. On the other hand, relying on the honesty of the RA for reputation soundness seems inevitable, since a malicious RA could always register additional phony users (i.e., Sybil identities) to arbitrarily create votes and inflate reputations.

At this point, one might raise the concern that, if each user has received a unique number of votes, the reputation value itself is identifying. Clearly, there is an inherent tradeoff between the precision of a measure of reputation and the anonymity of a user with any specific value, as pointed out by Steinbrecher [6]. The solution is to use a sufficiently coarse-grained reputation. When producing a signature in our construction, a user may prove any desired *lower bound* on their reputation instead of revealing the actual value. In this way, our construction allow users to implement their own policies for the precision of their reputations. For example, one policy would be to always round down to a power of two.

Algorithms. We now list and define the algorithms that constitute a scheme for signatures of reputation. All but VerifyRep may be randomized.

- $\textsc{Setup}(1^\lambda) \to (\mathsf{params}, \mathsf{authkey})$: The Setup algorithm is run once on security parameter 1^λ to establish the public parameters of the system params and a key $\mathsf{authkey}$ for the registration authority.
- $\textsc{GenCred}(\mathsf{params}, \mathsf{authkey}) \to \mathsf{cred}$: To register a user, the registration authority runs GenCred and returns the user's credential cred.
- $\textsc{GenNym}(\mathsf{params}, \mathsf{cred}) \to \mathsf{nym}$: The GenNym algorithm produces a one-time pseudonym nym from a user's credential.
- $\textsc{Vote}(\mathsf{params}, \mathsf{cred}, \mathsf{nym}) \to \mathsf{vt}$ or $\perp$: Given the credentials cred of some user and a one-time pseudonym nym, Vote outputs a vote from that user for the owner of nym, or $\perp$ in case of failure (e.g., if nym is invalid).
- $\textsc{SignRep}(\mathsf{params}, \mathsf{cred}, V, \mathsf{msg}) \to \Sigma$ or $\perp$: Given the credentials cred of some user, the SignRep algorithm constructs a signature of reputation Σ on a message msg using a collection of c votes $V = \{\mathsf{vt}_1, \mathsf{vt}_2, \ldots, \mathsf{vt}_c\}$ for that user. The signature corresponds to a reputation $c' \leq c$, where c' is the number of distinct users who generated votes in V. The SignRep algorithm outputs $\perp$ on failure, specifically, when V contains an invalid vote or one whose recipient is not the owner of cred.
- $\textsc{VerifyRep}(\mathsf{params}, \mathsf{msg}, \Sigma) \to c$ or $\perp$: The VerifyRep algorithm checks a purported signature of reputation on msg and outputs the corresponding reputation c, or $\perp$ if the signature is invalid.

The most basic property required of the above algorithms is *correctness*; we omit this definition for brevity. In the following section, we explore the other desired properties.

3 Privacy and Security Properties

The full version of this paper provides rigorous definitions for the four privacy and security properties [4]; here, we describe them at an intuitive level and discuss some of the subtleties in defining them appropriately.

Signer anonymity. First, we would like to ensure that a user may produce signatures of reputation anonymously. Furthermore, it should be impossible to determine whether two different signatures were produced by the same user. This may be defined by the following game. The challenger begins by generating the public parameters and a list of user credentials $\mathsf{cred}_1, \ldots, \mathsf{cred}_n$. An adversary $\mathcal{A}$ is given access to all the credentials and may use them to generate pseudonyms and votes before eventually printing a message msg, $1 \leq i_0, i_1 \leq n$, and two sets of votes V_0, V_1. The challenger flips a coin $b \in \{0, 1\}$ and returns $\Sigma_b = \textsc{SignRep}(\mathsf{params}, \mathsf{cred}_{i_b}, V_b, \mathsf{msg})$ to $\mathcal{A}$, which then prints a guess b'. We say that $\mathcal{A}$ has won the game if $b = b'$ and $\textsc{VerifyRep}(\mathsf{params}, \mathsf{msg}, \Sigma_0) = \textsc{VerifyRep}(\mathsf{params}, \mathsf{msg}, \Sigma_1)$. That is, the value of b should affect neither the reputation values of the resulting signatures nor their validity. If the advantage (that is, the probability of winning the game minus one-half) of every PPT $\mathcal{A}$ is negligible in the security parameter, we say that the scheme is *signer anonymous*.

Receiver anonymity. Complementing the ability to produce a signature of reputation anonymously is the ability to receive the necessary votes anonymously. In this case, we require that a pseudonym generated by the GenNym algorithm reveal nothing about its owner in the absence of that user's credential. An adversary $\mathcal{A}$ playing the corresponding game will select two users $1 \leq i_0, i_1 \leq n$ and must guess which produced the challenge $\mathsf{nym}^* = \textsc{GenNym}(\mathsf{params}, \mathsf{cred}_{i_b})$. Since we allow users to identify their own pseudonyms, we cannot provide all the credentials to $\mathcal{A}$ in this case. Instead, we provide $\mathcal{A}$ with access to an oracle which will reveal individual credentials on demand (a "corrupt" query) or use them to produce pseudonyms, votes, and signatures as requested. Then, to win the game, we require that $\mathcal{A}$ not corrupt either i_0 or i_1. We also require that $\mathcal{A}$ not request a signature from i_0 or i_1 using a vote that was cast for nym^*, since the reply would immediately reveal b (the signer is i_b iff the reply is not $\perp$). If the advantage of every PPT $\mathcal{A}$ in this game is negligible in the security parameter, the scheme is *receiver anonymous*.

Astute readers may note that we have not properly defined what it means for a vote to have been "cast for nym^*", since we have no information about how the adversary may have constructed it. To resolve this definitional issue, in the full version of this paper, we define "opening" algorithms which reveal the creator of a pseudonym and the voter and recipient of a given vote. To operate, they require a special opening key which may be generated during setup, just as in group signature schemes. However, while this tracing is an explicit feature of group signatures, here we use it only to establish a "ground truth" for definitional purposes. In an actual implementation, the opening key would not be generated.

Voter anonymity. We wish to define the voter anonymity property to encompass the strongest form of unlinkability compatible with the general semantics of the

scheme, as we did in the case of receiver anonymity. Doing so is more subtle in this case, however, due to the necessity of detecting duplicate votes. Because we require a SignRep algorithm to demonstrate the number of votes from *distinct* users, such an algorithm can be used by a vote receiver to determine whether two votes cast for any of their pseudonyms were produced by the same voter (duplicates). That is, the receiver can try to use the two votes to produce a signature and then check the reputation of the result with VerifyRep.

In defining voter anonymity, we allow precisely this type of duplicate detection, but nothing more. While initially this may seem like an "exception" to the unlinkability of votes, in actuality, it is not only inevitable,[1] but also unlikely to be a practical concern. Although a vote receiver must be able to detect duplicate votes, we can still avoid the voting histories we originally set out to eliminate. In particular, our definition ensures that in the following cases it is not possible to determine whether two votes were cast by the same user (i.e., to link the votes):

1. A user cannot link a vote for one of their pseudonyms with a vote for a pseudonym of another user, nor can they link two votes for distinct pseudonyms of another user (or two different users).
2. A *colluding group* of users cannot link votes between their pseudonyms, provided the pseudonyms correspond to different credentials. Furthermore, they are not able to link the *numbers* of duplicates they have observed. For example, if a user determines that they have received two votes from one user and three votes from another, they will have no way of matching these totals up with those of another colluding user.

In the corresponding game, $\mathcal{A}$ selects $1 \leq i_0, i_1 \leq n$ and nym and is given $\mathsf{vt}^* = \textsc{Vote}(\mathsf{params}, \mathsf{cred}_{i_b}, \mathsf{nym})$ as a challenge. As before, they are given access to the oracle and must make a guess b'. In this case, we require that if $\mathcal{A}$ requests through the oracle that the user corresponding to nym produce a signature using vt^*, then votes from both i_0 and i_1 must be included. Otherwise, the number of distinct votes in the resulting signature would directly reveal b. Additionally, we disqualify $\mathcal{A}$ if they both corrupt the user corresponding to nym and request a vote on nym from either i_0 or i_1. This is necessary because the status of such a vote as a duplicate of vt^* (or lack thereof) would reveal b. If every PPT $\mathcal{A}$ has negligible advantage in this game, the scheme is *voter anonymous*.

Reputation soundness. To define the soundness of a scheme for signatures of reputation, we use a computational game in which an adversary $\mathcal{A}$ must forge a signature of reputation Σ on some message msg. We disqualify $\mathcal{A}$ if Σ was the reply to one of its oracle queries, and we require that Σ have reputation strictly greater than what it could if the adversary had used the scheme normally. The value of the best such legitimately obtainable reputation will depend on several things: the number of users the adversary has corrupted (since the adversary

[1] Allowing proofs of vote distinctness while eliminating the ability to identify duplicates could only be possible if the notion of discrete votes is abandoned. This approach would require *all* votes in the system to be aggregated into a indivisible block before they can be used to produce signatures, a vastly impractical solution.

may use their credentials to produce votes), the number of votes received from honest users via oracle queries, and how those votes were distributed amongst the corrupted users. More precisely, let ℓ_1 be the *number of corrupted users* and ℓ_2 be the *greatest number of distinct honest users that voted for a single corrupt user*. Then we require that $\textsc{VerifyRep}(\mathsf{params}, \mathsf{msg}, \Sigma) > \ell_1 + \ell_2$ for the adversary to succeed. If, for every PPT $\mathcal{A}$, the probability of winning this game is negligible in the security parameter, then the scheme is *sound*.

In some applications, a weaker version of soundness may suffice and may be desirable for greater efficiency. One natural way to relax the definition is to specify an additional security parameter $\varepsilon \in (0, 1)$ as a multiplicative bound on the severity of cheating we wish to prevent. Specifically, we say that a scheme is *ε-sound* if it satisfies the above definition, but using the requirement that $(1 - \varepsilon) \cdot \textsc{VerifyRep}(\mathsf{params}, \mathsf{msg}, \Sigma) > \ell_1 + \ell_2$.

4 Highlights of Our Construction

Our scheme for signatures of reputation can produce sound signatures of reputation c of size $O(c)$ or ε-sound signatures of size $O(\frac{1}{\varepsilon} \log c)$. In the full version of this paper, we detail the construction and prove that it satisfies all of the properties discussed in the previous section [4]. In this section, we describe some of the scheme's technical features and underlying ideas.

Assumptions. Our constructions rely on a bilinear map (symmetric or asymmetric) between prime order groups. Its privacy and security properties are based on the relatively standard DLinear and SDH assumptions, BB-HSDH and BB-CDH [11], and a new constant-size, non-interactive, computational assumption called SCDH, which we prove hard in generic groups. Additionally, the ε-sound variant of our scheme requires the random oracle model.

Nested NIZKs. Throughout our construction, we make extensive use of the Groth-Sahai scheme for non-interactive zero-knowledge (NIZK) proofs [16], which can be used to efficiently demonstrate possession of signatures, ciphertexts, and their relationships while maintaining unlinkability properties. One unique (to our knowledge) feature of our construction is the use of *nested NIZKs*, that is, NIZKs which prove knowledge of other NIZKs and demonstrate that they satisfy the verification equations. This situation arises because a user's credentials contain a signature from the registration authority, and a user includes a NIZK proof of the validity of this signature when they cast a vote. When a signer later uses the vote, they include this NIZK within a further NIZK to demonstrate the validity of the votes while maintaining signer anonymity.

Proving distinctness. We give signers to ability prove the distinctness of their votes through the following mechanism. Each user credential contains (among other components) a "voter key" v and a "receiver key" r. A valid vote must contain a certain deterministic, injective function of these keys: $f(v, r)$. Thus, duplicate votes can be detected when $f(v_1, r) = f(v_2, r)$. To receive votes anonymously, a user includes in each nym an encryption of their receiver key $E(r)$ under their own public key. Using a homomorphism, the voter can use this ciphertext

to compute $E(f(v,r))$ and place it within the vote; later, the receiver will decrypt this to obtain $f(v,r)$. To maintain signer anonymity when using a series of votes $U_1 = f(v_1,r), U_2 = f(v_2,r), \ldots$ to sign a message, the signer blinds the votes with a (single) exponent to produce a list $U_1^s, U_2^s, \ldots$, which is included in the signature of reputation along with proof of knowledge of the exponent. Note that $U_1^s, U_2^s, \ldots$ will be distinct if the original values were.

Short signatures. To reduce the size of the signatures, we employ a sampling technique. Specifically, we can achieve ε-soundness while only including a random subset of the votes of size $O(\frac{1}{\varepsilon})$, *independent* of the original number of votes. To ensure the sample is random, we require the signer to first commit to the entire list of votes, then use the commitment as a challenge specifying which must be included. To efficiently demonstrate that the correct votes were included, we compute the commitment using a Merkle hash tree and include the corresponding off-path hashes with each vote, resulting in a final signature of size $O(\frac{1}{\varepsilon} \log c)$.

References

1. Narayanan, A., Shmatikov, V.: Robust de-anonymization of large sparse datasets. In: IEEE Symposium on Security and Privacy. (2008)
2. Backstrom, L., Dwork, C., Kleinberg, J.M.: Wherefore art thou r3579x? In: International World Wide Web Conference. (2007)
3. Arrington, M.: AOL proudly releases massive amounts of user search data. TechCrunch News (August 2006)
4. Bethencourt, J., Shi, E., Song, D.: Signatures of reputation: Towards trust without identity. `http://www.cs.berkeley.edu/~bethenco/sigrep-full.pdf`
5. Steinbrecher, S.: Enhancing multilateral security in and by reputation systems. In: FIDIS/IFIP Internet Security and Privacy Summer School. (September 2008)
6. Pingel, F., Steinbrecher, S.: Multilateral secure cross-community reputation systems for internet communities. In: TrustBus. (2008)
7. Steinbrecher, S.: Design options for privacy-respecting reputation systems within centralised internet communities. In: Intl. Information Sec. Conf. (SEC). (2006)
8. Belenkiy, M., Chase, M., Erway, C., Jannotti, J., Kupcu, A., Lysyanskaya, A., Rachlin, E.: Making p2p accountable without losing privacy. In: WPES. (2007)
9. Camenisch, J., Hohenberger, S., Lysyanskaya, A.: Balancing accountability and privacy using e-cash. In: Security and Cryptography for Networks (SCN). (2006)
10. Androulaki, E., Choi, S.G., Bellovin, S.M., Malkin, T.: Reputation systems for anonymous networks. In: Privacy Enhancing Technologies. (2008)
11. Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H.: Randomizable proofs and delegatable anonymous credentials. In: Crypto. (2009)
12. Camenisch, J., Kohlweiss, M., Soriente, C.: An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In: PKC. (2009)
13. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A.: Non-interactive anonymous credentials. In: TCC. (2008)
14. Groth, J.: Non-interactive zero-knowledge arguments for voting. In: ACNS. (2005)
15. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Secure distributed key generation for discrete-log based cryptosystems. In: Eurocrypt. (1999)
16. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Eurocrypt. (2008)