EARS: Enabling Private Feedback Updates in Anonymous Reputation Systems

Vishnu Teja Kilari, Ruozhou Yu, Satyajayant Misra, Guoliang Xue

***Abstract*—Reputation systems, designed to remedy the lack of information quality and assess credibility of information sources, have become an indispensable component of many online systems. A typical reputation system works by tracking all information originating from a source, and the feedback to the information with its attribution to the source. The tracking of information and the feedback, though essential, could violate the privacy of users who provide the information and/or the feedback, which could both cause harm to the users' online well-being, and discourage them from participation. Anonymous reputation systems have been designed to protect user privacy by ensuring anonymity of the users. Yet, current anonymous reputation systems suffer from several limitations, including but not limited to a) lack of support for core functionalities such as feedback update, b) lack of protocol efficiency for practical deployment, and c) reliance on a fully trusted authority. This paper proposes EARS, an anonymous reputation system that ensures user anonymity while supporting all core functionalities (including feedback update) of a reputation system both efficiently and practically, and without the need of a fully trusted central authority. We present security analysis of EARS against multiple types of attacks that could potentially violate user anonymity, such as feedback duplication, bad mouthing, and ballot stuffing. We also present evaluation of the efficiency and scalability of our system based on implementations.**

***Index Terms*—Reputation system, privacy, anonymity**

I. Introduction

Many online forums such as Stack Overflow, TripAdvisor and Yelp utilize reputation systems to evaluate the quality of information posted by users, while online marketplaces, such as Amazon and eBay employ reputation systems to ensure product quality and establish trust between sellers and buyers.

A typical reputation system works by tracking and publicizing information along with the reputation of its sources, feedbacks regarding it from other sources, and in many cases the reputation of the feedback sources as well. To achieve this, the system maintains a record of all users including both information providers and feedback providers, along with the reputation score that is intrinsically tied to each user's identity.

Let us consider how a typical reputation system works at present. When a user Alice wants to post information, the system creates a post and associates Alice's username and her reputation score to the post. This score plays a key role in determining the effect of the post, *e.g.*, in visibility and scope. Later when a user Bob wants to give feedback to this post, the reputation system also associates that feedback with Bob's username and his reputation score. The reputation system continuously tallies the feedback received for Alice's post and updates Alice's reputation score. All the subsequent posts by Alice are associated with this new reputation score. The role of reputation in determining the effect of the post and the role of the feedback in determining the new reputation score depend on the specific algorithms used in the reputation system.

One significant problem of such a system is the violation of privacy of user's identity by the system or other users of the system. Let us consider the following scenario. A user, Eve, provided a negative feedback on a post of Alice. Another user, Charlie, provided positive feedback on the same post. The current reputation systems let Alice learn the identities of Eve and Charlie (either directly or indirectly), which enables Alice to take biased actions, such as leaving unwarranted positive feedback for some or all of Charlies' posts and/or unwarranted negative feedback on some or all of Eves' posts. In extreme cases, Alice may take legal actions against Eve or cause physical and/or monetary harm to Eve. Another scenario involves other users or the system learning sensitive information about a user by tracking all the posts and feedbacks of that user. For example, based on the posts and feedbacks, a user's political inclination can be inferred [1]. Both these problems can have a huge impact on users' participation in these systems.

The goal of this paper is to address the privacy issues of existing reputation systems, which are present in several operational steps including registration, information posting, feedback posting, and feedback update. To ensure user anonymity and protect user privacy, a reputation system should function without relying on the true identities of the users. Further, the system should be general, practical, and scalable, in order to attain wide adoption in various real-world scenarios.

A. Motivation

Pseudonyms were traditionally regarded as an effective solution to the privacy concerns in online systems such as reputation systems. Unfortunately, Minkus *et al.* [2] showed that pseudonyms are not enough for identity anonymity via an attack that successfully exposed sensitive purchase histories of eBay users through analysis of transactions anonymized with pseudonyms. Before our work, many solutions have been proposed to address privacy in reputation systems [3]–[8]. Yet we find that none of the previously proposed solutions to this problem is sufficiently general, scalable, or practical for real world use. For instance, a number of systems [7], [8]

Kilari and Xue ({vkilari, xue}@asu.edu) are with Arizona State University, Tempe, AZ 85287. Yu (ryu5@ncsu.edu) is with North Carolina State University, Raleigh, NC 27606. Misra (misra@cs.nmsu.edu) is with New Mexico State University, Las Cruces, NM 88003. This research was supported in part by NSF grants 1704092, 1717197, 1719342, 1345232, 1914635, and EPSCoR Cooperative Agreement OIA-1757207 and DOE SETO award ED-EE0008774. The information reported here does not reflect the position or the policy of the funding agencies.

mandate verified usage of the concerned information or service by a user before the user can provide feedback (a user has to buy a product or use a service to provide feedback). These systems are not generic, but are more tailored towards online systems such as marketplaces or exchanges. Other limitations of existing works include dependence on a fully trusted central authority [3], [6], or vulnerability to attacks such as ballot stuffing [5].

A fair number of existing works rely on the existence of a fully trusted central authority [3], [6], which is violated when the authority can benefit from private user information, a situation inline to what happens in reality (e.g., Google and Amazon's handling of our data). Zhai *et al.* [4] proposed an anonymous reputation system based on the *anytrust* assumption (only an arbitrary one among a set of authorities is trusted). However, the system operates as a series of message and feedback rounds that may can take an arbitrary long period of time to complete. This makes the system impractical since a feedback phase might last perpetually long.

Another limitation of existing works is the lack of support for feedback update. In practice, the need of revising feedbacks is very important in many systems, *e.g.*, when a buyer of a product who initially gave a positive feedback wants to modify it because of a later found problem with the product. Existing works fail to find a way to address this issue because of the seemingly incompatibility between providing full anonymity to users and still maintaining user states to allow future updates. However, we believe that such a functionality is crucial to practical adoption of an anonymous reputation system due to its apparent need in many real-life scenarios, such as in the aforementioned marketplace example.

Based on these, our goal is to design a practical anonymous reputation system which enables feedback update in addition to all supported functionalities of existing reputation systems (including user registration, information posting and feedback), while still protecting the anonymity and privacy of the benign users from malicious users and a curious system operator.

B. Our Contributions

We propose EARS, a secure, practical, and scalable anonymous reputation system that achieves our goal, without relying on any fully trusted central authority. In EARS, every information or feedback post by a user will be associated with a unique name and a unique reputation score. This unique score is (less than but) close enough to the actual score so that it reflects the true reputation of a user, yet far enough that the user cannot be tracked using it, thus protecting user anonymity and privacy.

EARS tallies the feedback received for a post and provides a way to anonymously update the reputation of the user who posted the information. EARS also enables users to provide or change their feedback anonymously on a post anytime in the future. EARS does not track users' historical activities or build long term identities. Neither EARS nor its users can link different posts/feedback of a user, or learn the identity of a user through its reputation score. Another advantage of EARS is that it enables weighted feedback. In other words, since the reputation of the user providing feedback is included in the feedback (vote), this reputation can be used in order to determine the weight (effect) of the feedback. These are realized utilizing known cryptographic primitives, including blind signatures [9] and partially blind signatures [10], whose security has been rigorously studied in the literature.

To summarize, the **main contribution** of this paper is the design of EARS, a secure, scalable and practical anonymous reputation system that implements all required functionalities (registration, information posting, feedback posting, and feedback update) while ensuring the anonymity and privacy of all users, along with detailed security analysis and implementation-based performance evaluation of the proposed system.

The rest of the paper is organized as follows. In Section II, we present our system and threat models. In Section III, we describe the details of EARS. In Section IV, we analyze the security of EARS. In Section V, we present our experiments and results. In Section VI, we discuss the related work. In Section VII, we present our conclusions and future work.

II. System and Threat Models

A. System Model

We assume EARS is implemented on a server controlled by the system operator. In our system, many users can interact with EARS. Since each user interacts with EARS independently, we describe and evaluate the performance of EARS in terms of the interactions between a specific user and the system. However, since each user's actions may affect other users (*e.g.*, when a malicious user is trying to break the privacy of others), our security analysis in Section IV takes into account all users in the system. A user may hold one or both of two independent roles: as a ***poster*** who provides information, and/or as a ***voter*** who provides feedback. We describe the relevant protocols for these two roles independently. In practice, multiple servers may be used for load-balancing or back-up. Since the security of our system does not rely on having multiple servers, we assume only one server in our narrative for brevity.

For illustration, we use P to represent a user posting some information (poster), V to represent a user providing feedback on some information (voter), and S to represent the EARS system, respectively. A Public Key Infrastructure (PKI) is used by EARS and all users. Note that a decentralized PKI can be used [11], [12], which waives the need for relying on a trusted central authority in system setup. The system S is assigned with (and identified by) one public-private key pair (pu, pr). Each user performs account initialization when joining the system, which associates the user identity ID, its initial reputation r, and a credential token $h(ID||r)_{pr}$ (blindly) signed by S. All the communications happen over established secure channels which ensure the confidentiality and integrity of the communications.

B. Threat Model

We model the server deploying EARS as *semi-honest* (*a.k.a.*, honest-but-curious [13]), and the users of EARS as *malicious*. The EARS server follows the defined protocol but will attempt

TABLE I
TABLE OF NOTATIONS

$h(m)$	Hash of the message m
$(m)_k$	Signature on message m using key k
ϕ^V	Voting token
τ^P	Posting token
π^P	Feedback collection token
γ^V	Feedback update token
$b_e(m)$	Blinding factor e applied to message m
$u_e(m)$	Unblinding factor e applied to message m
$BS(m)$	Blind Signature on m
$PBS(m)_x$	Partial Blind Signature on m with value x
(pu, pr)	Public/Private key pair of EARS

to learn all possible information from messages received. The assumption of a semi-honest system is realistic due to the various practical factors that prevent the system operator from deviating from the pre-defined protocols, including oversight from regulating authorities, the huge potential cost at stake, and the fear of reputation damage. Users can be malicious as they can deviate from the protocol, collude among themselves, or try to violate the anonymity of other users by tracking reputation scores, pseudonyms, and reputation updates within EARS.

We assume all adversaries are computationally bounded. Public key encryption, symmetric key encryption, and hash functions are correctly implemented. We also assume that the network and upper-layer protocols do not leak the users' identifiable information to the EARS. In order to thwart traffic analysis attacks, we assume that the network connections between the users and EARS are established over anonymous communication channels such as Tor [14]. The works on *k*-anonymization [15] observe that content of the information itself is enough to identify the source if additional information about the source is available through auxiliary data sources. This is orthogonal to the problem addressed by this paper. In Table I, we summarize the notations used in the paper.

III. DESIGN OF THE EARS PROTOCOLS

In this section, we discuss the detailed protocol design of EARS. A user can be a poster (information provider) and/or a voter (feedback provider), and we assume that they have already initialized their credentials with the system. Fig. 1 illustrates the protocols followed by a poster, voter, and the system (EARS). These protocols include: poster registration, voter registration, posting, voting, reputation update, and feedback update. Fig. 1(a) shows three protocols that a poster uses to interact with the system: registration, posting, and reputation update. Fig. 1(b) shows the three protocols for the voter: registration, voting, and feedback update.

In poster registration, a poster provides her credentials to the server, and requests an anonymous posting token certifying her identity, reputation, and intention to post, along with an anonymous feedback collection token that is coupled with this post to enable feedback attribution. The tokens are only issued after the credentials of the poster are verified by the system.

In the posting protocol, the poster provides the posting token along with the information she wishes to post. The system verifies that the posting token is valid, and then proceeds to post the information given by the poster along with the reputation of the poster enclosed in the posting token. The poster token was blinded during registration to ensure the anonymity of the poster during the posting protocol.

Similar to poster registration, in the voter registration protocol, a voter intending to provide feedback proves her credentials and requests an anonymous voting token certifying her identity, reputation, and intention to vote on a specific post, along with an anonymous feedback update token that is coupled with this voting token to enable feedback update. The server verifies the credentials, and issues the anonymous voting token and the anonymous feedback update token to the voter.

Then, during the voting protocol, the voter provides the voting token along with the feedback she wishes to post. The system verifies the voting token, and then posts the feedback given by the voter along with the reputation of the voter, which is enclosed in the voting token. The blinded voting token protects the anonymity of the voter during the voting protocol.

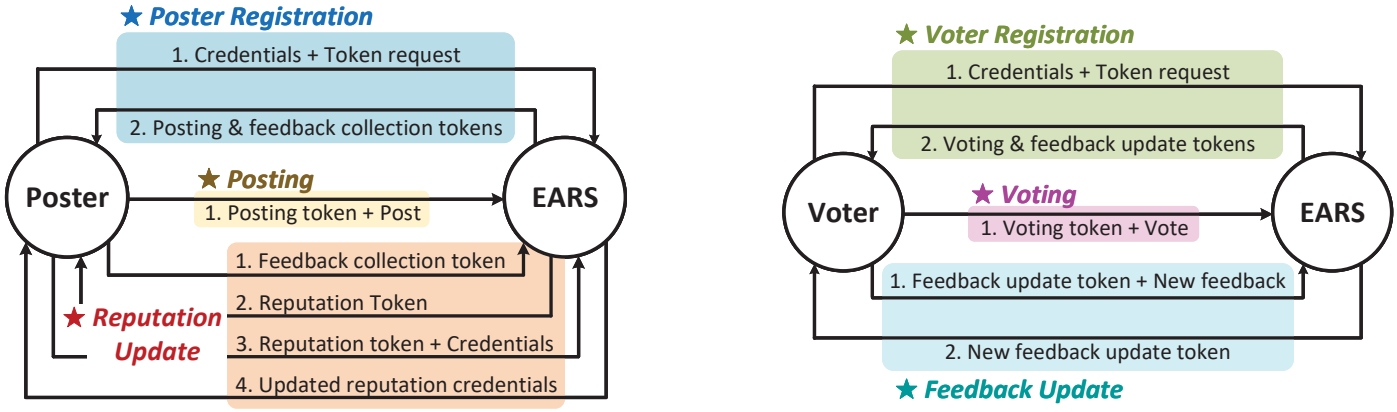
A poster executes the reputation update protocol to collect feedback to its post, by submitting the feedback collection token to the system. After verifying the feedback collection token, the system tallies all the feedback received by the post until then and issues a partially blind signature on the reputation token which is an anonymous token containing the feedback score and the timestamp. This reputation token is only redeemable by the poster who posted the information. Any subsequent changes to the feedback will result in issuance of a new anonymous reputation token reflecting the changes to the feedback score.

After receiving partially blind signature of the system on the reputation token, the poster applies the unblinding factor to it and retrieves the signed reputation token. This token must be redeemed within a week (specified by the timestamp) and when the poster redeems this token, she submits this signed reputation token to the system along with her credentials. The system verifies that the credentials are valid and the time to redeem the token has not expired and updates the reputation score of the poster based on the reputation token and issues new credentials to the poster reflecting the updated reputation score. In the feedback update protocol, the voter will submit the feedback update token issued during voter registration along with the updated feedback. The system verifies the token and if the token is valid, the system updates the feedback of a post corresponding to the token with the new feedback.

In the following, we explain these protocols in detail.

A. Poster Registration Protocol

Protocol 1 shows the registration protocol, which is used to verify a poster's credentials and credibility and issue an anonymous posting token and an anonymous feedback collection token to the poster by S. In Step 1, a poster P sends S its identity (ID_P), a reputation score (r), the credential ($h(ID_P||r)_{pr}$) and a credibility token ($h(ID_P||t)_{pr}$) where t is the number of reputation tokens redeemed during previous week. This credibility token is issued by the system to the poster after verifying that the number of reputation tokens redeemed by



(a) Poster protocols: Poster Registration, Posting, and Reputation Update

(b) Voter protocols: Voter Registration, Voting, and Feedback Update

Fig. 1. Illustration of protocol interactions between user and EARS.

the user is same as the number of posting tokens issued to her and all the reputation tokens are redeemed recently. This can be easily verified using the timestamp on the reputation tokens. The poster also sends a blinded posting token, $b_k(a_k, r_k, h(a_k||r_k), B_k(h(ID_P)), h(a_k||B_k(h(ID_P))))$ and a value p. The term (b_k, B_k) corresponds to the blinding values; a_k is a random nonce; r_k corresponds to a reputation score less than the actual reputation score r; and p is a tag informing the system that this is a request for a posting token by the user (to perform posting). Then, S verifies the validity of the arguments inside the blind posting token using cut-and-choose protocols. The verification using the cut-and-choose protocols involves P sending n blind tokens to S. Each of the n blind tokens will have unique nonces, unique reputation scores less than the actual reputation score, and unique blinding values.

Protocol 1 Registration Protocol for Poster P

Input: ID, r, $h(ID_P||r)_{pr}$, and $h(ID_P||t)_{pr}$ of poster P.
Output: Poster P acquires a posting token and a feedback collection token from S

1: P sends its credentials along with blind tokens to S:
$P \rightarrow S$: $ID_P, r, h(ID_P||r)_{pr}, p, h(ID_P||t)_{pr}$
$b_k(a_k, r_k, h(a_k||r_k), B_k(h(ID_P)), h(a_k||B_k(h(ID_P))))$.
2: P sends the feedback collection token to S:
$P \rightarrow S$: $b_{n+1}(x), b_{n+2}(h(a_k||x))$.
3: After S verifies the blind token, it issues a partially blind signature on a posting token (k^{th} token) and a blind signature on the feedback collection token to P:
$S \rightarrow P$: $PBS(b_k(a_k, r_k, h(a_k||r_k), B_k(h(ID_P)),$
$h(a_k||B_k(h(ID_P))))_p, BS(b_{n+1}(x)),$
$BS(b_{n+2}(h(a_k||x)))$.

In Step 2, after S ensures that arguments in the blind token from P are valid using the cut-and-choose protocol, P also sends feedback collection token, $b_{n+1}(x)$ and $b_{n+2}(h(a_k||x))$. In Step 3, S issues a partially blind signature on the posting token, $PBS(b_k(a_k, r_k, h(a_k||r_k), B_k(h(ID_P)), h(a_k||B_k(h(ID_P))))_p$, and a blind signature on the feedback collection token, $BS(b_{n+1}(x)), BS(b_{n+2}(h(a_k||x)))$. The poster P applies the unblinding factors to retrieve the signature of S on the posting token which is denoted as $\tau_i^U = PBS(a_k, r_k, h(a_k||r_k), B_k(h(ID_P)), h(a_k||B_k(h(ID_P))))_p$, and the feedback collection token which is denoted as $\pi_i^U = BS(x), BS(h(a_k||x))$.

The poster registration protocol can be executed offline since it does not require any details regarding the information to be posted. It can also be executed multiple times in advance to collect multiple tokens. Since the tokens are anonymously obtained and unlinkable, accumulation of tokens will not violate the anonymity of the poster. While posting the information, the poster can select one of the unused tokens and submit the information to be posted along with it.

B. Posting Protocol

Protocol 2 presents the procedure for a poster to anonymously post information using an obtained posting token. As described in Table I, $PBS(q)_y$ denotes the partially blind signature on q, with y as the mutually agreed upon value by both the requester (P) and the issuer (S) of the partially blind signature. In Step 1, P presents the posting token $PBS(a_k, r_k, h(a_k||r_k), B_k(h(ID_P)), \ h(a_k||B_k(h(ID_P))))_p$ to S along with values a_k,r_k,$B_k(h(ID_P))$ and $post_k$ where a_k, r_k and $B_k(h(ID_P))$ are the random nonce, reputation score and blinded-hash of the identity (which serves as the reputation token) respectively, and $post_k$ is the information to be posted. Then, S compares its signature on the entities a_k, r_k and $B_k(h(ID_P))$ with the submitted token, and if verified, posts the information $post_k$ with a unique identity id_k and the reputation score r_k. This identity id_k denotes the identity of the post and not the identity of the poster who posted this post.

C. Voter Registration Protocol

Protocol 3 details the procedure for a voter V to obtain from S an anonymous voting token to provide feedback for a post of identity id_l, and an anonymous feedback update token for potential feedback update in the future.

In Step 1, V presents to S its identity ID_V, and reputation r_V along with the credential $h(ID_V||r_V)_{pr}$ accompanied by a blinded voting token $b_l(a_l, r_{Vl}, h(a_l||r_{Vl}))$ and a

Protocol 2 Posting Protocol for Poster P

Input: Posting token issued by S to the poster P.
Output: Poster P posts a message anonymously.

1: P sends its credentials a_k, r_k, the message $post_k$ and $B_k(h(ID_P))$ along with the posting token to S:
$P \rightarrow S$: $a_k, r_k, post_k, B_k(h(ID_P))$,
$PBS(a_k, r_k, h(a_k||r_k))$,
$B_k(h(ID_P)), h(a_k||B_k(h(ID_P))))_p$.
2: S posts message $post_k$ with id_k and reputation score r_k.

bit v (tag signifying intent to vote) and the identity of the post it wants to vote on, id_l. Here, b_l corresponds to the blinding value, a_l is a random nonce and r_{Vl} is a reputation score less than V's actual reputation score r_V. The system S uses the cut-and-choose protocol to ensures the validity of the arguments. The verification using the cut-and-choose protocols involves V sending m blind tokens to S. Each of the m blind tokens will have unique nonces, unique reputation scores less than the actual reputation score, and unique blinding values. In Step 2,after S ensures that arguments in the blind token from V are valid using the cut-and-choose protocol, V sends the feedback update token, $b_{m+1}(z)$, $b_{m+2}(h(a_l||z))$. In Step 3, S issues a partially blind signature on the voting token, $PBS(b_l(a_l, r_{Vl}, h(a_l||r_{Vl})))_{(v,id_l)}$, and a blind signature on the feedback update token, $BS(b_{m+1}(z))$, $BS(b_{m+2}(h(a_l||z)))$. The voter V applies the unblinding factors to retrieve the signature of S on the voting token, which is considered as ϕ_i^V, $\phi_i^V = PBS(a_l, r_{Vl}, h(a_l||r_{Vl}))_{(v,id_l)}$, and then applies unblinding factors to the feedback update token, which is considered as γ_i^V, $\gamma_i^V = BS(z), BS(h(a_l||z))$.

Protocol 3 Registration Protocol for Voter V

Input: ID_V, r_V, and $h(ID_V||r_V)_{pr}$ of the voter V.
Output: V obtains voting token and feedback update token.

1: V sends its credentials along with blind tokens to S:
$V \rightarrow S$: $ID_V, r_V, h(ID_V||r_V)_{pr}, v, id_l$
$b_l(a_l, r_{Vl}, h(a_l||r_{Vl}))$.
2: V sends the feedback update token to the S:
$V \rightarrow S$: $b_{m+1}(z), b_{m+2}(h(a_l||z))$.
3: After S verifies the blind token, S issues a partially blind signature on voting token (l^{th} token) and a blind signature on the feedback update token to V:
$S \rightarrow V$: $PBS(b_l(a_l, r_{Vl}, h(a_l||r_{Vl})))_{(v,id_l)}$,
$BS(b_{m+1}(z)), BS(b_{m+2}(h(a_l||z)))$.

D. Voting Protocol

Protocol 4 explains the procedure followed by the voter V to provide anonymous feedback for a post with identity id_l using the voting token acquired from S. In Step 1, V presents the voting token $PBS(a_l, r_{Vl}, h(a_l||r_{Vl}))_{(v,id_l)}$ to S along with values a_l, r_{Vl} and vote, $vote_l$ where a_l is the random nonce, r_{Vl} is the reputation score, and $vote_l$ is the feedback of the voter V for id_l. Then, S compares its signature on the entities a_l and r_{Vl} with the submitted token, and after verifying that the signature is valid, it assigns $vote_l$ to the post associated with the id_l along with the reputation score of V, r_{Vl}.

Protocol 4 Voting Protocol for Voter V

Input: Voting token issued by S to the voter V.
Output: Voter V votes for a posting anonymously.

1: V sends its credentials a_l and r_{Vl} along with the signed blind tokens to S:
$V \rightarrow S$: $a_l, r_{Vl}, vote_l, PBS(a_l, r_{Vl}, h(a_l||r_{Vl}))_{(v,id_l)}$.
2: S validates signature, and posts $vote_l$ for post id_l and the reputation score r_{Vl} of the voter.

E. Reputation Update Protocol

In the reputation update protocol, a poster interacts with S to update its reputation based on feedback it has received from voters on its posts. The poster submits the feedback collection token of the post to S. After verifying the token, S tallies all the feedback received by the post and issues a partially blind signature on the reputation token, which is an anonymous token containing the current feedback score of the poster and the timestamp, and sends it to the poster. This token is redeemable only by the poster who posted the information. Only one token for a post is issued per update period (*e.g.*, a week, decided by the system operator). Subsequent changes to the feedback will result in issuance of a new anonymous reputation token reflecting the changes to the feedback score. This protocol is executed by the poster for each of its posts every update period.

Protocol 5 explains the procedure followed by S to update the reputation of the poster P based on the feedback (votes) its post id_k has received. Here, id_k is the identity of the post and S does not know the identity of P or the voters that voted on the post; thus their anonymity is preserved throughout the process. In Step 1, P presents to S the feedback collection token $\pi_i^U = BS(x), BS(h(a_k||x))$ (obtained during poster's registration for the post) along with the credentials in the token

Protocol 5 Reputation Update Protocol for Poster P

Input: Feedback collection token issued by S to poster P.
Output: Updated reputation score of the poster P with the feedback f corresponding to the post id_k.

1: P sends the feedback collection token along with the credentials in the token to S:
$P \rightarrow S$: $a_k, x, BS(x), BS(h(a_k||x))$.
2: S verifies arguments and its signature on the token and issues a partial blind signature on the reputation token:
$S \rightarrow P$: $PBS(B_k(h(ID_P)))_{(f,w)}$.
3: P sends the new reputation token to S and requests a new feedback collection token:
$P \rightarrow S$: $b_c(c), b_{c+1}(h(a_k||c)), B_c(h(ID_P))$.
4: S replaces the reputation token with the new reputation token and blind signs the new feedback collection token:
$S \rightarrow P$: $BS(b_c(c)), BS(b_{c+1}(h(a_k||c)))$.
5: P submits the signature of S on reputation token along with its credentials:
$P \rightarrow S$: $ID_P, r, h(ID_P||r)_{pr}, PBS(h(ID_P))_{(f,w)}$.
6: S verifies received credentials and signature and issues updated reputation score and credentials to P:
$S \rightarrow P$: $ID_P, r+f, h(ID_P||r+f)_{pr}$.

a_k, x. The system S signs x and matches it with $BS(x)$, checks the posts associated with the value a_k, computes $h(a_k||x)$ and compares its signature on $h(a_k||x)$ with the submitted token $(BS(h(a_k||x)))$. If all the credentials and the tokens submitted by P are valid, then P is the poster of this information. In Step 2, S performs a partially blind signature on the reputation token $(B_k(h(ID_P)))$ of the post associated with a_k with the feedback f (the total feedback received for this post) and the time stamp in terms of week (the week out of 52 weeks), w and sends this partially blind signature, $PBS(B_k(h(ID_P)))_{(f,w)}$ to P. Here, the time stamp w is added to ensure that the reputation update protocol will be executed for each post regularly. On receiving the partial blind signature, P applies the unblinding factor and retrieves the signature on the reputation token $(PBS(h(ID_P))_{(f,w)})$.

In Step 3, P sends the new reputation token to S and requests a new feedback collection token for the next (future) reputation update. Then, S updates the post associated with a_k with the new reputation token and issues a new feedback collection token to P in Step 4. In Step 5, to update his/her reputation, P sends its credentials along with the signed reputation token to S. In Step 6, S verifies the credentials and its signature on the reputation token and updates the reputation score r of P associated with the identity ID_P with feedback f and issues updated reputation score and corresponding credentials to P.

F. Feedback Update Protocol

Feedback update is one of the core functionalities that we need to support in an anonymous reputation system. Protocol 6 details the procedure followed by V to update her feedback on a post identified with the identity id_l. Voter V sends the feedback update token she obtained during voter registration, along with the credentials in the token and the updated feedback $(vote_{updated})$: $a_l, z, BS(z), BS(h(a_l||z)), vote_{updated}$ to S in Step 1. S verifies that its signature on z matches with the feedback update token submitted $(BS(z))$, checks the vote associated with the value a_l, computes $h(a_l||z)$, and compares its signature on the $h(a_l||z)$ with the submitted token $(BS(h(a_l||z)))$. If all the credentials and the tokens submitted by V are valid, then V is the voter who provided this feedback. S then updates the old feedback $vote$ with the new feedback $vote_{updated}$. In Step 3, V requests a new feedback update token from S for a future feedback update, which S issues in Step 4.

Protocol 6 Feedback Update Protocol for Voter V

Input: Feedback update token issued by S to the voter V.
Output: Updated feedback of the voter V on the post id_l.

1: V presents the blindly signed feedback update token, the credentials in the token, and the updated feedback to S: $V \rightarrow S$: $a_l, z, BS(z), BS(h(a_l||z)), vote_{updated}$.
2: S verifies the argument and the signature on the token, and updates the vote associated with a_l on id_l by V.
3: V requests a new feedback update token from S: $V \rightarrow S$: $b_d(d), b_{d+1}(h(a_l||d))$.
4: S issues a new feedback update token to V: $S \rightarrow V$: $BS(b_d(d)), BS(b_{d+1}(h(a_l||d)))$.

IV. Security Analysis

In this section, we perform detailed security analysis of EARS.

Proposition 1: *Poster anonymity is guaranteed in EARS.* □

During poster registration, the poster P sends her credentials ID_P, r, and $h(ID_P||r)_{pr}$ along with the credibility token, $h(ID_P||t)_{pr}$ to the anonymous reputation system S. Then, S verifies the submitted credentials with its database of users, checks the validity of the credibility token and if they match, S knows that P is authentic. Poster P sends n blinded tokens; each token is composed of the following components that are unique to each token: a random nonce; a reputation value less than P's actual reputation; a commitment with the nonce and the reputation value; a reputation token; and the commitment of the reputation token with the nonce. S uses cut-and-choose protocol to verify these messages, and then issues a partially blind signature on one of the tokens with p.

The security of partially blind signature ensures that S cannot know the nonce, reputation value, feedback collection token, and their commitments, which are signed by it for the kth token despite unblinding the other $(n-1)$ tokens. The blind signature on the feedback collection tokens also ensures that these values are concealed from S when it issues its signature on them. So, S issues an anonymous blind token which contains valid arguments (reputation score, feedback token, and feedback collection token) to an authenticated poster. In the posting protocol, P sends the anonymous blind token (with the signature of S) along with the arguments in the token: a_k, r_k, $B_k(h(ID_P))$ to S.

Since the signature itself can only be verified and cannot be compared to anything in the past (due to the application of unblinding factors to the partially blind signature), S cannot know the identity of the poster to which this signature was issued. The arguments do not leak any information regarding the identity of the poster because the nonce is random, the reputation score is less than the original reputation score (which is not known by the system), and the blinded hash is a result of a cryptographic irreversible one-way function.

During the reputation update protocols, P presents the signature of S on the feedback collection token along with the arguments in it: a_k, x. Then, S verifies that the signature is valid on the submitted arguments. Since this signature is obtained by unblinding the blind signature on the feedback collection token, S cannot connect this signature or the arguments with the identity of the poster to which this signature was issued. The reputation token $(B_k(h(ID_P)))$ itself was blinded with blinding factor B_k, preventing S from knowing the identity of the poster to which this token was going to be issued. The signature of S on this token is a partially blind signature with feedback score f and timestamp w. When P redeems this reputation token, S can keep track of all the reputation tokens of value f it issued and try to correlate that with the posters that redeemed those tokens. Since many posters might have been issued the reputation tokens of value f, deanonymizing the identity of a specific poster using only the value of reputation

tokens is difficult (especially for large anonymity sets).

Proposition 2: *Voter anonymity is guaranteed in EARS.* □

In the voter registration protocol, a voter V sends her credentials ID_V, r_V, and $h(ID_V||r_V)_{pr}$ to the system S. Then, S verifies the submitted credentials as with the poster. Then V sends m blinded tokens; each token is composed of the following components that are unique to each token: a random nonce, a reputation value less than the actual reputation value of the voter, and a commitment with the nonce and the reputation value. The system uses cut-and-choose protocol to verify that all these arguments are valid, and then S issues a partially blind signature on one of the tokens with (id_l, v) (indicating that the token is for voting on a post, id_l). Since it is a partially blind signature, S cannot know the nonce, reputation value, and their commitments it had signed.

The system cannot infer these values from the values obtained from $(m-1)$ unblinded tokens because they are unique to each token. The blind signature on the feedback update token also ensures that these values are concealed from S when it issues its signature on them. So, S issues an anonymous blind token which contains valid arguments (reputation score, and feedback update token) to an authenticated voter V. During the voting protocol, V sends the anonymous blind token (signature of S on its token) and also the arguments in the token: a_l, r_{Vl} to S. Since the signature itself can only be verified and not compared to anything stored in the past (due to the application of unblinding factors to the partial blind signature), S cannot know the identity of V.

The arguments do not leak any information regarding the identity of the voter because the nonce is random, and the reputation score is less than the original reputation score (which is not known by the system). The vote or the identity of the post that is being voted on using this token cannot identify the voter. So, S receives only a signed token, which contains arguments necessary to vote on a specific post anonymously – the voter is not identifiable. In the feedback update protocol, V presents the signature of S on the feedback update token along with the arguments in it: a_l, z. Then, S verifies that the signature is valid on the submitted arguments. Due to the blind signature on the feedback update token, S cannot connect this signature or the arguments of the token with V's identity.

Proposition 3: *EARS cannot link the various tokens of the poster or the voter.* □

Since the tokens issued by the system S to a user (a poster or a voter) are always anonymous, S cannot differentiate between the tokens issued to the same user or a different user. As such, the system has no information regarding the various tokens of a user (poster/voter) to link them.

Proposition 4: *A poster cannot use higher reputation score of another poster for posting.* □

The commitment of the random nonce (used to post the information) with the reputation score during the poster registration protocol prevents a poster from using a higher reputation score of another poster to post. Only the poster who obtained a token with her identity and a reputation score less than her reputation score will be able to use the token issued during poster registration protocol to post some information. Reputation scores cannot be created unless the poster interacts with S, further the several interactions between P and S end up functioning as challenge-response ensuring that a simple reputation of somebody else cannot be replayed or stolen and used. Even if another poster obtains this token and posts some information, the hash of the identity of the poster (who obtained the token) in the reputation token will ensure that reputation update based on the feedback for the post will only occur for the poster whose token has been used to post.

Proposition 5: *A poster can only redeem his/her feedback token once.* □

The hash of the identity of the poster (who posted the information) in the feedback token will ensure that no other poster can redeem this feedback token. The unique signature of the system on this feedback token will prevent double spending of this token – once a feedback token is redeemed, the system S keeps track of it so that it cannot be redeemed again.

Proposition 6: *EARS is immune to bad mouthing and ballot stuffing.* □

Since EARS enables weighted feedback, the reputation will play a vital role in determining the effect of the feedback, which in turn decides the reputation of the information source. As such, unless the reputation of a malicious voter is significant enough or a large group of voters collude, the effect of authentic feedback will outweigh the fake feedback. Thus, attempts on bad mouthing and/or ballot stuffing are not viable.

Non-goals: Our system cannot detect the malicious users but is resilient against such users. Like many previous works on anonymous reputation systems, our system is not immune to sybil attacks but it mitigates their effect. Our system cannot defend against network-level DoS and DDoS attacks.

V. Evaluation

In this section, we discuss the implementation of EARS and analyze the results to demonstrate that it is efficient, robust, scalable, and can be implemented without any additional hardware or software requirements. We implemented EARS using standard crypto libraries in Java. To emulate real world hardware, a user (poster/voter) is implemented on an Intel Core i5-2450M, 2.5 GHz machine with 8 GB RAM. To emulate EARS, we implemented the server on an Intel Core i7-6700K, 4.0 GHz machine with 32 GB RAM. All the results were averaged over 1000 runs. As we mentioned, the interaction of a user (poster/voter) with EARS is independent of interactions of other users with EARS – the system can interact with multiple users in parallel. This would ensure that EARS is scalable, since increase in the number of users would only increase the computation requirements of EARS linearly.

For better scalability, in EARS more servers can be added to serve an increasing number of clients with low latency. We measure the time taken by the user and the server for each protocol in our system. We consider only the computation steps and not the data transmission steps as they are dependent on the nature and traffic in the network. We also do not consider the storage requirements because they are trivial (in the order

of several kilobytes for each client). Fig. 2 shows the time taken at the user during the poster registration protocol (PRP), the voter registration protocol (VRP), the reputation update protocol (RUP), and the feedback update protocol (FUP). Fig. 3 shows the time taken at the server during PRP, VRP, the posting protocol (PP), the voting protocol (VP), RUP, and FUP.

As shown in Fig. 2, the user part of the poster registration protocol takes only 22.9 ms ($n = 10$). It involves creating the blind tokens which in turn involve hashing, and applying blinding factors. After receiving the posting token, the poster applies the corresponding unblinding factor to retrieve the signature of the system on the posting token and the feedback collection token. The user part of the voter registration protocol takes only 12.8 ms ($m = 10$). It involves creating the blind tokens which involve hashing and applying blinding factors. Since the voter registration protocol does not involve reputation token, the number of blinding and hashing operations are less compared to the poster registration protocol. This is the reason that the time required for the voter registration protocol is less than that of the poster registration protocol. After receiving the

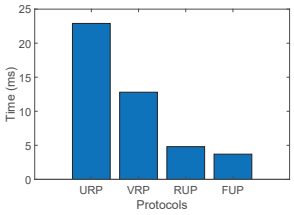


Fig. 2. Time taken by a user in various protocols of our system.

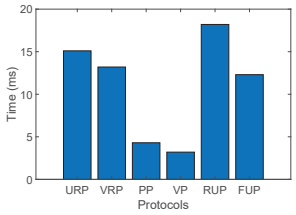


Fig. 3. Time taken by the server in various protocols of our system.

the feedback update token, the poster applies the corresponding unblinding factor to retrieve the signature of the system on the feedback update token. The user part of the posting protocol and the voting protocol are not mentioned because there are no computation steps for the users in these protocols. They submit the tokens (which are signed by the system) and the arguments in the tokens along with the posts or votes to the system. The users do not need to perform any computations in these protocols. So, they are omitted from Fig. 2.

The reputation update protocol takes 4.8 ms; this is because it only requires a few blinding and unblinding operations. The feedback update protocol requires the voter to submit the feedback update token along with the arguments in it and the updated feedback (vote) to the system. The voter also applies blinding factor to the new feedback update token request. After receiving the blind signature on the new feedback update token, it applies unblinding factor to retrieve the new signed token. The user takes 3.7 ms to complete feedback update.

As shown in Fig. 3, the server part of the poster registration protocol takes 15.1 ms. It encompasses the unblinding and verification of tokens in cut-and-choose protocols which involves hashing. After verifying that the tokens are valid, the system performs a partially blind signature on the posting token and a blind signature on the feedback collection token and send them to the poster. The server part of the voter registration protocol takes 13.2 ms. It constitutes the unblinding and verification of tokens in cut-and-choose protocols which encompasses hashing. After verifying that the tokens are valid, the system issues a partially blind signature on the voting token and a blind signature on the feedback update token. The amount of time taken by the system for the poster registration protocol is more than the voter registration protocol because of the presence of the reputation token, which increases the number of hashing and verification operations.

TABLE II
EXECUTION TIME FOR A SINGLE SERVER WITH VARYING CLIENTS

# Clients	VRP	PP	VP	FUP
10	88.7 ms	23.9 ms	23.6 ms	92.8 ms
100	743.5 ms	215.8 ms	212.3 ms	848.1 ms
1000	6811.9 ms	2059.6 ms	2047.7 ms	8191.9 ms

In the posting protocol, the system verifies its own signature on the posting token and verifies the arguments in the posting token with the arguments submitted to it by the poster, which involves hashing operations. The server takes 4.3 ms to execute the posting protocol and 3.2 ms for the voting protocol. The voting protocol involves the system verifying its own signature on the voting token and verifying the arguments inside that token with the arguments submitted to it by the voter, which involves a hashing operation. In the reputation update protocol, the server takes 18.2 ms. It requires several complex operations, such as verifying signature on the feedback collection token and integrity of the arguments inside the token, and issuance of a partially blind signature on the reputation token.

After receiving a request for a new feedback collection token, the system issues a blind signature on the new feedback collection token. When the poster submits a reputation token for redemption, the system verifies its signature on the reputation token and the integrity of the arguments inside the token. After verifying, the system updates the reputation score of the poster and creates new credentials representing the updated reputation score of the poster and sends these credentials to the poster. This step involves signature of the system on the hash of the identity of the poster and new reputation score. In the feedback update protocol, the server takes 12.3 ms. The system part of the feedback update protocol involves the system verifying its signature on the feedback update token submitted by the voter and the integrity of its arguments. It also involves issuing blind signature on the new feedback update token submitted by the voter. As mentioned previously, the poster registration protocol and the reputation update protocol can be executed offline. The voter registration protocol, the posting protocol, the voting protocol and the feedback update protocol are the only protocols that need to be executed in real time.

As shown above, the voting protocol and the posting protocol take very little time to execute. The maximum amount of time taken among them is 4.3 ms. A processor core of a server running these protocols can service 200 users simultaneously in less than a second in real time. Use of cloud computing can easily enable our system to scale economically and efficiently. The other protocols which can be executed offline can be run during the off-peak hours of the system. Table II shows the time taken by the server to simultaneously serve varying number

of users during the various online protocols of the system. The poster registration and the reputation update protocols are not shown in Table II, because they can be executed offline. The experimental results from the implementation of EARS demonstrate its efficiency and scalability.

VI. Related Work

We discuss several proposed solutions for anonymous reputation systems and make a case for the need of our system. Anonymous reputation systems in P2P networks based on electronic cash (e-cash) have been proposed [16], [17]. Anonymity in e-cash is essential to ensure the unlinkability of users' transactions. The main drawback of e-cash based anonymous reputation systems is their inability to support negative feedback. This drawback makes these systems untenable for many application scenarios since negative feedback plays a vital role in calculating the reputation of a user. Another drawback of these systems is their inability to provide different levels of granularity for feedback and reputation.

Blind signatures based anonymous reputation systems have also been proposed [3], [18]. These systems leverage blind signatures to hide the identity of users performing transactions. The drawback of these systems is that they trust a central authority. Wang *et al.* [3] proposed blind signatures for anonymous reputation and trust in participatory sensing, but this scheme trusts the server to issue the feedback tallies. These tallies can be used by the server to identify the participants. Blömer *et al.* [6] proposed a group signature based anonymous reputation system. But this system trusts an entity (group manager) to be honest. Schaub *et al.* [5] proposed a blockchain based trustless reputation system. This system is prone to ballot stuffing attacks because the service provider issues the credentials which are used by the customers to issue feedback.

Some anonymous reputation systems only allow a one-time feedback per access to the service to be evaluated. For instance, Soska *et al.* [8] proposed a decentralized anonymous marketplace with secure reputation. The system requires the clients to purchase products from the vendor to be able to leave feedback. While this property of requiring the feedback provider to perform a prior action is useful in some scenarios, it is not deployable in platforms such as Stack Overflow, TripAdvisor, and Yelp where there is no service to be accessed. There are many platforms on the web apart from the marketplaces which rely on reputation systems to function effectively.

Zhai *et al.* [4] proposed a tracking resistant anonymous reputation system using verifiable shuffles, linkable ring signatures, and homomorphic cryptography. This system follows the anytrust model. In order to scale efficiently and effectively, the system makes a tradeoff between security and the efficiency after a threshold of users. Another drawback of this system is that it operates in a series of message and feedback rounds which lasts for an arbitrary amount of time (with finite time interval) based on the application scenario. This makes the system impractical for realistic deployment since the feedback phase might be perpetual for many real-world applications.

VII. Conclusions and Future Work

In this paper, we proposed a secure, practical and scalable anonymous reputation system named EARS. EARS allows anonymous information (post) posting and feedback (voting) by users. In both cases, the reputation of the user is associated with the post/vote it submits to the system. After receiving the feedback on a post, our system allows the reputation of the posting user to be updated with the feedback received for that post. This reputation update is also anonymous. In case a voter decides to update her feedback on a specific post, our system enables her to do so anonymously as well. Detailed security analysis of our system shows that it is immune to attacks from malicious users or any attempt to violate the anonymity of the user by the system. We note that our system is not fully immune to sybil attacks, which is an orthogonal problem to what we solved. However, since our system enables weighted feedback, such a feature can be leveraged to mitigate the effect of sybil accounts. We will tackle sybil attacks in our future work.

References

[1] T. Islam, A. R. Bappy, T. Rahman, and M. S. Uddin, "Filtering political sentiment in social media from textual information," in *Proc. IEEE ICIEV*, 2016, pp. 663–666.

[2] T. Minkus and K. W. Ross, "I know what you're buying: Privacy breaches on ebay," in *Proc. PETS*, 2014, pp. 164–183.

[3] X. O. Wang and *et al.*, "Artsense: Anonymous reputation and trust in participatory sensing," in *Proc. IEEE INFOCOM*, 2013, pp. 2517–2525.

[4] E. Zhai, D. I. Wolinsky, R. Chen, E. Syta, C. Teng, and B. Ford, "Anonrep: towards tracking-resistant anonymous reputation," in *Proc. USENIX NSDI*, 2016, pp. 583–596.

[5] A. Schaub, R. Bazin, O. Hasan, and L. Brunie, "A trustless privacy-preserving reputation system," in *Proc. IFIP SEC*, 2016, pp. 398–411.

[6] J. Blömer, J. Juhnke, and C. Kolb, "Anonymous and publicly linkable reputation systems," in *Proc. FC*, 2015, pp. 478–488.

[7] A. Kokoschka, R. Petrlic, and C. Sorge, "A reputation system supporting unlinkable, yet authorized expert ratings," in *Proc. ACM SAC*, 2015, pp. 2320–2327.

[8] K. Soska, A. Kwon, N. Christin, and S. Devadas, "Beaver: A decentralized anonymous marketplace with secure reputation." *IACR Cryptology ePrint Archive*, vol. 2016, p. 464, 2016.

[9] D. Chaum, "Blind signatures for untraceable payments," in *Springer Advances in Cryptology*, 1983, pp. 199–203.

[10] T. Okamoto, "Efficient blind and partially blind signatures without random oracles," in *Proc. IACR TCC*, 2006, pp. 80–99.

[11] B. Qin, J. Huang, Q. Wang, X. Luo, B. Liang, and W. Shi, "Cecoin: A decentralized pki mitigating mitm attacks," *Elsevier Future Generation Computer Systems*, 2017.

[12] C. Fromknecht, D. Velicanu, and S. Yakoubov, "A decentralized public key infrastructure with identity retention." *IACR Cryptology ePrint Archive*, vol. 2014, p. 803, 2014.

[13] A. Paverd, A. Martin, and I. Brown, "Modelling and automatically analysing privacy properties for honest-but-curious adversaries," *Tech. Rep.*, 2014.

[14] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *Proc. USENIX Security*, 2004, pp. 303–320.

[15] L. Sweeney, "k-anonymity: A model for protecting privacy," *World Scientific International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.

[16] E. Androulaki, S. G. Choi, S. M. Bellovin, and T. Malkin, "Reputation systems for anonymous networks," in *Proc. PETS*, 2008, pp. 202–218.

[17] J. Camenisch, S. Hohenberger, and A. Lysyanskaya, "Balancing accountability and privacy using e-cash," in *Proc. SCN*, 2006, pp. 141–155.

[18] D. Christin, C. Roßkopf, M. Hollick, L. A. Martucci, and S. S. Kanhere, "Incognisense: An anonymity-preserving reputation framework for participatory sensing applications," *Proc. IEEE PerCom*, vol. 9, no. 3, pp. 353–371, 2013.