Guo L, Zhang C, Fang Y *et al.* A privacy-preserving attribute-based reputation system in online social networks. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 30(3): 578–597 May 2015. DOI 10.1007/s11390-015-1547-9

A Privacy-Preserving Attribute-Based Reputation System in Online Social Networks

Linke Guo [1] (郭霖珂), *Member, ACM, IEEE*, Chi Zhang [2] (张 驰), *Member, IEEE*
Yuguang Fang [3,*] (方玉光), *Fellow, IEEE*, and Phone Lin [4] (林 风), *Senior Member, ACM, IEEE*

[1] *Department of Electrical and Computer Engineering, Binghamton University, Binghamton, NY 13902, U.S.A.*

[2] *CAS Key Laboratory of Electromagnetic Space Information, University of Science and Technology of China, Hefei 230026, China*

[3] *Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, U.S.A.*

[4] *Department of Computer Science and Information Engineering, "National" Taiwan University, Taipei 10617, China*

E-mail: lguo@binghamton.edu; chizhang@ustc.edu.cn; fang@ece.ufl.edu; plin@csie.ntu.edu.tw

Received July 15, 2014; revised January 8, 2015.

Abstract Online social networks (OSNs) have revolutionarily changed the way people connect with each other. One of the main factors that help achieve this success is reputation systems that enable OSN users to mutually establish trust relationships based on their past experience. Current approaches for the reputation management cannot achieve the fine granularity and verifiability for each individual user, in the sense that the reputation values on such OSNs are coarse and lack of credibility. In this paper, we propose a fine granularity attribute-based reputation system which enables users to rate each other's attributes instead of identities. Our scheme first verifies each OSN user's attributes, and further allows OSN users to vote on the posted attribute-associated messages to derive the reputation value. The attribute verification process provides the authenticity of the reputation value without revealing the actual value to entities who do not have the vote privilege. To predict a stranger's behavior, we propose a reputation retrieval protocol for querying the reputation value on a specific attribute. To the best of our knowledge, we are the first to define a fine-grained reputation value based on users' verified attributes in OSNs with privacy preservation. We provide the security analysis along with the simulation results to verify the privacy preservation and feasibility. The implementation of the proposed scheme on current OSNs is also discussed.

Keywords reputation, privacy, attribute, authentication

1 Introduction

1.1 Background

It has been witnessed that online social networks (OSNs) provide many features for people nowadays, such as content sharing, instant messaging, and making new friends, which help people move their daily life to the cyberspace. As an important component of OSNs, reputation systems offer users the ability to vote on and collect the reputation in order to build up social status and/or selectively make new friends, e.g., excluding users with bad reputation. To vote on and obtain certain reputation, people may use different types of identities, or even pseudonyms to communicate, which obvious lowers the credibility of the values as well as the OSNs. In another word, without a real identity or any verified information, the authenticity of users and

Regular Paper

This work was partially supported by the National Science Foundation of USA under Grant No. CNS-1423165. The work of Chi was supported in part by the National Natural Science Foundation of China (NSFC) under Grant Nos. 61202140 and 61328208, the Program for New Century Excellent Talents in University of China under Grant No. NCET-13-0548, and the Innovation Foundation of the Chinese Academy of Sciences under Grant No. CXJJ-14-S132. Lin's work was supported in part by MoE ATU Plan, the Taiwan Science and Technology Authority under Grant Nos. MOST 103-2622-E-009-012, MOST 103-2221-E-002-152-MY3, MOST 103-2221-E-002-249-MY3, MOST 104-2923-E-002-005-MY3, and MOST 103-2627-E-002-008, and the ICL/ITRI Project of Chunghwa Telecom.

*Corresponding Author

their reputation values cannot be guaranteed. To tackle this issue, some OSNs apply real identities for the reputation management. For example, LinkedIn, which provides detailed education background, working experiences, and real social relationships, helps people look for jobs and professional occupations. People prefer to use real identity information to increase opportunities for jobs, and to "endorse" other users' skill that could help them obtain certain reputation in specific areas. However, the revealing of users' profile detail associated with real identity information in the cyberspace would compromise users' privacy. Meanwhile, for the definition of reputation values, some of the existing reputation systems in OSNs only offer each individual user an overall reputation value based on his/her behavior in the system. However, it is known that the general reputation value on a particular user cannot clearly represent one's real reputation. For example, Bob may have a good reputation as a car mechanic but not as an expert on the stock market. On the other hand, as an objective reputation system, users' reputations should be obtained based on what users do, e.g., their posted messages or comments, instead of who they are in general. Hence, users' reputation values on different aspects should be built upon different contents that they post on OSNs, such that we can obtain the fine-grained and objective reputation instead of a blurring and non-specific one. Therefore, it is highly desirable to design a fine-grained content-based reputation system for OSNs, where users' real identities are hidden from the reputation management procedures. In the proposed reputation system, a user's reputation value is an aggregated value which depends on other users' ratings of the user's posted messages.

1.2 Related Work

Trust and Reputation System. A thorough survey[1] on reputation systems for online service provision describes the current trends and development in this area. Cho *et al.*[2] discussed the trust together with the reputation management for mobile ad hoc networks. There are various ways to define reputation or trust value in the literature[3-9]. Lin *et al.* proposed a peer-to-peer architecture for heterogeneous social networks in [10], which allows users from different types of social networks to communicate. The proposed architecture also highlights the reputation managements in different types of professional social networks. The most relevant work to our proposed system is [11], where Benthencourt *et al.* proposed a novel cryptographic primitive *signature of reputation* which supports monotonic measures of reputation in a complete anonymous setting. Their proposed scheme eliminates the procedure when users want to verify an identity. Instead, the verification of the signature directly reveals the signer's reputation, where the user's identity privacy has been preserved. However, since their scheme purely relies on two end users, it can only support monotonic reputation. Moreover, their scheme lacks efficiency due to the nested non-interactive zero-knowledge proofs.

e-Cash and e-Voting Schemes. Lin *et al.* in [12] proposed an efficient secure trading model named mobile electronic payment (MEP) for wireless mobile networks, which applies the ID-based cryptography for key agreement and authentication. The work of Androulaki *et al.* in [13] is close to ours in its aims. However, directly applying the e-cash based approaches would not prevent users from inflating the reputation by giving multiple votes. Another line of research in this area would be e-voting schemes[14-15] proposed by Groth, in which it allows the voting while maintaining certain privacy properties. However, it deviates from our design purposes in two ways: first, they do not consider the vote receiver's privacy issue; second, the fine-grained reputation design is beyond the scope of those studies. We compare the existing studies whose goals are similar to our design in Table 1.

Anonymous Credentials. We apply several techniques originated from schemes of anonymous credentials[16-17] in terms of setting up the trust from a centralized trust authority. Unfortunately, those schemes lack a mechanism for proving that votes or credentials

Table 1. Comparison Among Major Studies

	Fine-Grained	Non-Monotonic	Voter Anonymity	Receiver Anonymity	Double-Voting	Distinct Votes	Trust Authority
Benthencourt *et al.*[11]			√	√	√	√	
Androulaki *et al.*[13]			√	√			
Groth *et al.*[14-15]			√	√	√		
Camenisch *et al.*[16]			√		√		√
Our approach	√	√	√	√	√	√	√

come from distinct users while simultaneously hiding identities of users.

Zero-Knowledge Proof System. For NIZK, the first work was introduced by Blum *et al.* in [18]. Our scheme employs the non-interactive proof system for bilinear pairing[19] which has been used for several applications in [20-25]. Unfortunately, we cannot directly apply the scheme in [19] due to different scenarios and assumptions. For the zero-knowledge proof used in our scheme, we also cannot apply traditional encryption schemes that use shared secret to vote on a particular user. In most attribute-based encryption schemes[26-28], the key distribution center is responsible for distributing public/private key pairs based on each individual's attributes and the corresponding structure. If they are in the same attribute group, they may mutually authenticate each other. However, our proposed system cannot allow the vote receiver to share a voting privilege with voters before the voting.

1.3 Contributions

In this paper, we design a fine-grained attribute-based reputation system for online social networks, which has the following features.

1) *Verifiability and Anonymity.* Different from the systems which verify users' identity or offer users the ability to use pseudonyms, our proposed system is able to verify users' attributes and allows them to use verified information to vote on, collect, and publish the fine-grained reputation value based on each individual's posted content while maintaining the anonymity of users. The key ingredient of providing both verifiability and anonymity is the application of the zero-knowledge during the verification process.

2) *Fine-Grained Attribute-Based Reputation.* We first define the attribute-based reputation for each user in our proposed system, where users need to first get the user-centric attributes verified and then choose the attributes to establish the reputation. Rather than directly revealing the identity, the key advantage of our scheme is shifting the reputation established on the real identity to each user's attributes, such that users maintain their identity privacy while realizing the original functionality of reputation management.

3) *Authenticity of Reputation Value.* We only consider votes generated by voters who have the corresponding expertise on the particular attribute. Compared to the votes generated by general users, the opinion coming from the users with verified speciality would increase the credibility of reputation values.

To the best of our knowledge, we are the first to design the content-based reputation system which relies on users' attributes while preserving the privacy of the verifiability of attributes and authenticity of the reputation value.

The remainder of this paper is organized as follows. Section 2 introduces preliminaries for some cryptographic techniques used in our paper. We describe the system and adversary models in Section 3, along with the security objective. The proposed scheme is presented in detail in Section 4, followed by the performance analysis in Section 5. Finally, Section 6 concludes the paper.

2 Preliminaries

2.1 Global Parameters for Proposed System

2.1.1 Group Definition

We define a group for the Pedersen commitment scheme and RSA-based signature in our proposed system. Let the public parameters be a group $\hat{G}$ of prime order p, and a set of generators $\hat{G} = (\hat{g}) = (\hat{g}_1,\ \hat{g}_2, ...)$. We assume the strong RSA assumption[29] is hard. Define $\hat{n} = pp'$, where $p = 2p_1 + 1$, $p' = 2p_2 + 1$ are safe primes and $p = p' = \Theta(2^k)$, where k is the security parameter. For the element $\hat{g} \in \hat{G}$, $\hat{g}$ is a quadratic residue modulo $\hat{n}$, where $\hat{n}$ is a special RSA modulus of $2k$ bits.

Pedersen Commitment. In order to commit to a block of message $\hat{m} = (\hat{m}_1, \hat{m}_2, ..., \hat{m}_w) \in Z_p^w$, pick a random $r \in Z_p$ and $\hat{h} \in \hat{G}$ to compute the commitment as $\hat{C} = \hat{h}^r \prod_{i=1}^{w} \hat{g}_i^{\hat{m}_i}$.

2.1.2 Bilinear Pairing

Bilinear pairing operations are performed on elliptic curves[30]. Let G_1 and G_2 be groups of the same prime order p. Discrete logarithm problem (DLP) is assumed to be hard in both G_1 and G_2. Let P denote a random generator of G_1 and $e : G_1 \times G_1 \to G_2$ denote a bilinear map constructed by modified Weil or Tate pairing with the following properties:

1) Bilinear: $e(aP, bQ) = e(P, Q)^{ab}$, $\forall P, Q \in G_1$ and $\forall a, b \in Z_p^*$, where Z_p^* denotes the multiplicative group of Z_p, the integers modulo p. In particular, $Z_p^* = \{x \mid 1 \leqslant x \leqslant p-1\}$.

2) Non-degenerate: $\exists P, Q \in G_1$ such that $e(P, Q) \neq 1$.

3) Computable: there exists an efficient algorithm to compute $e(P, Q), \forall P, Q \in G_1$.

Bilinear pairing is the basic operation in the identity-based cryptosystem, the non-interactive witness-indistinguishable (NIWI) and zero-knowledge proofs (NIZK), all of which are used as the fundamental techniques in our scheme.

2.2 Zero-Knowledge Proof

NIWI *and* NIZK *Proof.* We apply part of the non-interactive proof system in [19], which gives a formal definition for both non-interactive witness-indistinguishable and zero-knowledge proof. We define $\mathcal{R}$ as a computable ternary relation. Given a tuple $(crs, st, wit) \in \mathcal{R}$, we call crs as the common reference string, st as the statement that we need to prove and wit as the witness. Note we also use $\mathcal{L}$ to denote the language consisting of statements in $\mathcal{R}$. Suppose $\mathcal{R}$ consists of three polynomial time algorithms $(\mathcal{K}, \mathcal{P}, \mathcal{V})$, where $\mathcal{K}$ is crs generation algorithm, $\mathcal{P}$ and $\mathcal{V}$ are prover and verifier, respectively. $\mathcal{P}$ takes a tuple (crs, st, wit) as input and outputs a proof π, while $\mathcal{V}(crs, \pi, st)$ will output 1 if the proof is acceptable and 0 if not acceptable. The proof system $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ should satisfy completeness and soundness properties, where completeness denotes if the statement is true, an honest verifier is convinced of this fact by an honest prover, and soundness shows that if the statement is false, and the cheating prover can convince the honest verifier that is true with a negligible probability. For NIWI, we require no adversary can distinguish the real crs and the simulated crs, while adversaries cannot distinguish which witness the prover uses. For zero-knowledge, we require no verifier obtain additional information other than the fact that the statement is true.

Zero-Knowledge Proofs about Discrete Logrithm. This paper also uses well-known techniques for proving statements about discrete logarithms, such as proof of knowledge of a discrete logarithm modulo a prime[31] and proof that a commitment opens to the product of two other committed values[32]. These protocols are secure under the discrete logarithm assumption. When referring to the proofs, we will use the notation introduced by Camenisch and Stadler[33]. For instance, $PK\{(r, s) : y = g^r h^s\}$ denotes a zero-knowledge proof of knowledge of exponents r and s such that $y = g^r h^s$ holds. All values in the parenthesis, in this example, r, s, denote quantities whose knowledge to be proven, while all other values are known to the verifier.

3 System Model

3.1 Overview

First of all, we give a high-level description of our proposed system. The attribute-based reputation of a user is an aggregated value based on other users' ratings, where those users perform as voters to give subjective values on whether messages posted by the user are useful or not. To achieve the fine-grained reputation values on different attributes, we need to guarantee the privacy of both the attribute and the identity during the protocol run. As shown in Fig.1, we consider the scenario where Bob posts an attribute-associated message, and every valid user in the system is able to view,

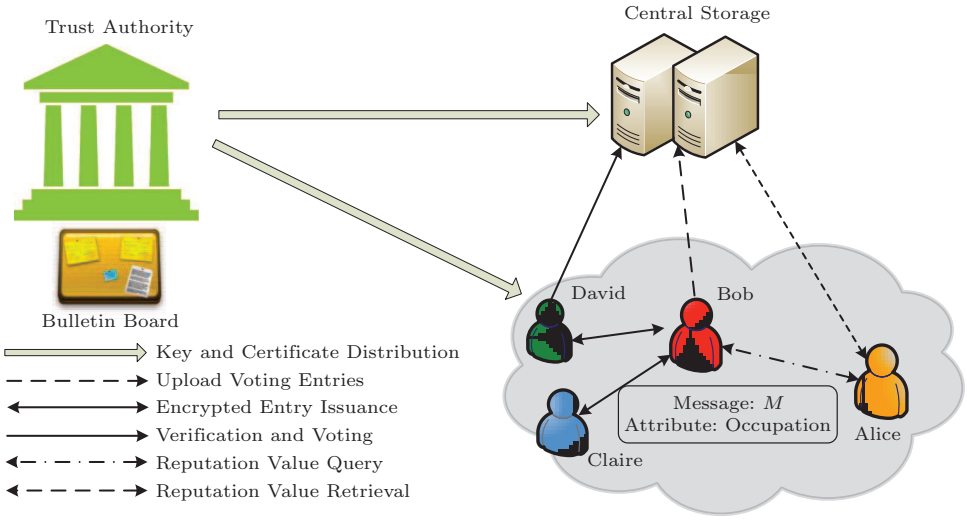


Fig.1. System model.

vote and retrieve the reputation on this item. The central storage is the same as the centralized infrastructure lays in the existing social networks. Here we define the attribute-associated message as a message which is associated with one or a set of verified attributes, e.g., Bob posts a message "F is a good car shop for the service on tires", which is bound with a specific attribute value *occupation: mechanic*. David, who is an expert on car mechanic, sends a voting request to Bob. Then, Bob returns the voting entries in the central storage to David, which enables David to use his verified attribute to vote on this attribute-associated message. The key ingredient of our proposed scheme is the non-interactive zero-knowledge on verifying David's designated attribute. Since David is in need of obtaining the certain entry for voting, he is required to verify by himself the valid and verified attributes to Bob while maintaining zero-knowledge to Bob. The central storage stores Bob's encrypted attribute in certain entries before the voting process. As soon as the central storage obtains David's vote, it first verifies the authenticity of the David's voting information and then checks the privilege that Bob gives to David. If all the checks pass, the central storage calculates the vote (verified attribute) together with the one stored in the central storage to obtain the results. To guarantee the authenticity of the reputation value, we require that the vote is valid only when both David's and Bob's attribute values are the same, or we consider the vote as an invalid one due to the fact that David is not qualified to give opinions based on his verified information. Another crucial issue that needs to be carefully considered is to avoid "double-voting", and we apply part of the e-cash scheme to prevent this malicious behavior. After the central storage obtains the results, querier Alice uses proxy re-encryption scheme to query the central storage for Bob's reputation value on a specific attribute. As Bob continues to use pseudonyms to publish messages, Alice can only obtain Bob's attribute-based reputation without knowing the real identity of Bob.

Some entities and related definitions of the system model are introduced as follows.

• *Trust Authority.* The trust authority (TA) in our proposed system is a fully-trusted infrastructure which is responsible for the generation of system parameters. Apart from that, TA also distributes the public/private key pairs for valid users based on their real ID as well as their pseudonyms key pairs which may be used for voting and collecting the reputation value. More importantly, TA verifies and issues the credentials and certificates corresponding to the user's attributes and values. Voters use these credentials and certificates to vote the message that the vote receiver posts online.

• *Central Storage.* We assume there is a semi-trusted central storage (CS) in our system. The centralized storage can easily be found in the existing online social networks, such as Facebook, Twitter, and LinkedIn. However, those centralized infrastructures store users' private information together with users' key pairs, which obviously compromises users' privacy. To the contrary, in our scheme, we only store the ciphertext and non-sensitive information in the central storage to avoid potential privacy leakage.

• *Users.* Different from traditional online social network service, our system enables TA to give not only the real identity to each user, but also a set of collision-resistance pseudonyms. Users may use their assigned pseudonyms to request the voting privilege, vote for a particular message, and retrieve the reputation value. More specifically, users in our system have different roles when they take different actions, where they can be the *voters* who intend to vote for a particular user, *vote receivers* who issue the voting privilege, or *queriers* who want to obtain a designated user's reputation.

• *Attribute and Attribute Value.* In this paper, we distinguish an attribute and its attribute value during the protocol run. An attribute is a description of a particular user in specific aspect, e.g., affiliation, gender, age, interests. However, to verify those attributes only does not reveal private or sensitive information. For example, if a user's occupation is verified by TA, showing the verified information will not expose the specific work or profession of this user. On the other hand, we define attribute value as a specific value of the corresponding attribute, such as *occupation: car mechanic*. For the authenticity of reputation value, we require the vote should be generated based on the verified attributes and the corresponding values.

• *Message Space.* Since our reputation system relies on the content posted by each user, we define the content as an attribute-associated message $\mathcal{M} \in Z_p$. Note that a message cannot be some sentences that are publicly known correct, like axioms or common sense. Except those, users are free to choose a message with the corresponding attribute, such as "Our service on tires is the best in town" with the attribute value *occupation: car mechanic* or "The camera we sell has good quality" on eBay with the attribute value *occupation: camera seller*. The voter would rate those messages based on whether they are helpful or accurate, and at-

tach the opinion together with the verified attributes as a vote.

• *Reputation Value.* We divide reputation value into two parts: one rated by common users and the other generated by users with verified professional. In this paper, we only consider the latter situation where the reputation value is coming from the one who has the professional expertise on the worthiness of a posted message. The user's attribute-based reputation value is a numerical value ranging between $[-N, N]$, where N is the total number of users. When the user performs as a voter, he/she uses his/her verified attribute and the corresponding value and chooses between $\{1, -1\}$ based on his/her judgement on a particular message, where 1 denotes agree and -1 disagree. The reputation value stored in the central storage shows the overall votes on the specific attribute of a user, while any valid user can obtain the reputation value by querying the central storage.

3.2 Security Objective

Our major security objective is to preserve the privacy of users' voting attributes and voters' identities during the protocol run. Since TA verifies user-centric attributes that they wish to establish the reputation in this OSN, we also cannot leak the credential of an attribute in a plaintext to any other party (users and the central storage). However, we allow the verification of an attribute without revealing the linkage between the identity and the corresponding attributes. Furthermore, the attribute information that a user implemented for verification cannot be identical. Otherwise, it is easy for the adversary, or even a benign user to trace back to that particular user. Users' identity privacy is also a crucial issue that needs to be carefully addressed. The identity privacy preservation is two-fold: *voter anonymity* and *vote receiver anonymity*. Firstly, when a voter intends to vote on a particular message published by a vote receiver, he/she should be guaranteed the *voter anonymity* that no one is able to trace him/her according to the particular vote. For the *vote receiver anonymity*, a vote receiver sometimes wants to publish something which may threaten the identity privacy or incur unintended issues if he/she uses real identity. Therefore, our scheme offers vote receiver anonymity which prevents identity privacy from being traced and leaked when vote receivers post some messages that they wish to receive vote for. More importantly, the central storage takes charge of collecting votes and it should maintain the indistinguishability to votes in terms of both values and voters, where we refer it as the confidentiality of the central storage. On the other hand, "double voting" on a specific message is strongly prohibited in our proposed scheme. Otherwise, a user is able to inflate his/her reputation by voting him/herself multiple times.

3.3 Adversary Model

We consider the following attacks to our proposed scheme. For the most possible active attacks to a voting scheme, the voter may "double vote" a designated vote receiver in order to maliciously inflate the reputation value. In addition, users may actively launch Sybil-attack to achieve their malicious purposes. On the other hand, users may intend to collect others' attribute information to launch impersonation attacks, which may be potentially used to spoof reputation values. Also, the disclosure of any user's attribute obviously compromises users' privacy. Another type of severe and complex adversary monitors a specific user's voting behaviors on different attributes, which may potentially leak the identity privacy of that user. Since our voting scheme relies on distinct messages, we prohibit the voter from using the voting privilege to vote messages to which it is not given the privilege. For example, if a user publishes two messages, we do not consider the scenario where the voter votes on the second message based on his/her previous decision on the first message. For passive attacks, since our system may be deployed in a wireless environment, an adversary may eavesdrop wireless communication channels or modify and inject bogus data during message transmissions. According to our assumption, the central storage is a semi-trusted entity, which is curious but honest, in the sense that we cannot disclose users' votes and attributes in a plaintext form. Furthermore, we consider the collusion attack among a group of users, where they use their valid and verified attributes to spoof one's reputation value. However, we will not consider the collusion attack launched by malicious users and central storage. We will not consider the possibility of sharing secrets with others either, since this type of active attacks cannot be prevented in most systems.

4 Proposed Scheme

In this section, we elaborate our privacy-preserving attribute-based reputation system in detail. The proposed system enables users in OSNs to anonymously

vote on, collect, and retrieve reputation values. We devise four major schemes to realize the basic functionalities of the proposed reputation system, which are attribute verification, attributed-based reputation voting privilege check, anonymous voting, and reputation value retrieval. To ease the reading, we have listed the major notations in Table 2.

Table 2. Main Notations

Notation	Description
ξ	Security parameter
e	Bilinear map of G_1
$f, h, \hat{f}, \hat{h}, \hat{z}$	Elements in G_1
$\mathcal{G}$, $\mathcal{F}$	Pseudo-random function
$\mathcal{PS}$	Pseudonym set
pk/sk	ID-based key pairs
vsk/vk	Vote key pair
x	User's attribute
$\mathcal{M}$	User's posted message
$\tilde{v}_\imath$	Credential on attribute $\imath$
$w_\jmath^*$	Plaintext of σ^{-1}
$\overline{c}, \overline{d}$	Commitments
$r, s, t, \hat{r}, \dot{r}$	Random numbers in Z_p^*
N	Total number of users
$\mathcal{A}$	User's attribute set
$p = \Theta(2^k)$	Order of the groups
$\hat{e}_6$, $\hat{e}_9$	Bilinear map of M_1
u_1, u_2, u_3	Elements in M_1
$\mathbb{P}$	Random permutation
PS_A^κ	User A's pseudonym
$\overline{pk}/\overline{sk}$	Selective-tag key pair
pks/sks	TA's signing key
$\tilde{x}_\imath$	TA's secret key on attribute $\imath$
$m_{\imath.\jmath}$	Value $\jmath$ of attribute $\imath$
$\sigma_{\imath.\jmath}$	Certificate on $\imath$'s value $\jmath$
c_j	Ciphertext after $\mathbb{P}$
$\pi, \hat{\pi}, \ddot{\pi}$	NIWI, NIZK proofs
ε, ϖ	Voting secret numbers
$L(x)$	Label of data x
σ_{TA}	TA's signature on (ε, ϖ)

4.1 System Setup

The system setup is mainly executed by TA to initialize the security domain for generating parameters and distributing identity/pseudonym key pairs to valid users in the system. We also briefly review the structured encryption scheme and the non-interactive proof system, both of which are used as major building blocks for our system.

4.1.1 Parameter Generation and Issuance

In the system setup, TA assigns the ID-based public/private key pairs for each user in the system, and then TA may go offline as we have assumed before. For the ease of description, we assume that there is a secure channel between TA and the user in the system, like SSL or TSL, which can be achieved by any public key cryptosystems. The key generation procedures are as follows[30]: 1) input the security parameter ξ to the system and output parameter tuple (p, G_1, G_2, e, g, H); 2) randomly select a domain master secret $\varsigma \in Z_p^*$ and calculate the domain public key as $P_{\mathrm{pub}} = \varsigma g$. Then, TA publishes the domain parameter tuple $(p, G_1, G_2, e, g, H, P_{\mathrm{pub}})$ and maintains ς confidentially, where $H(\cdot)$ is defined before as $H(\cdot) : \{0,1\}^* \to G_1$, and g is a generator of G_1. Given a specific public $ID \in \{0,1\}^l$, the public/private key (pk_{ID}/sk_{ID}) pair is $H(ID)/\varsigma H(ID)$, which is distributed by TA during the initiation process. We also assign a bunch of collision-resistant pseudonyms for anonymous communications. Taking user A as an example, it is given a set of pseudonyms, $\mathcal{PS}_A = \{PS_A^\kappa | 1 \leqslant \kappa \leqslant |\mathcal{PS}_A|\}$. Based on the pseudonyms, each pseudonym of A is given a set of secret keys as $sk_{\mathcal{PS}_A} = \{sk_{PS_A^\kappa}\} = \{\varsigma_A H(PS_A^\kappa) \in G_1 | 1 \leqslant \kappa \leqslant |\mathcal{PS}_A|\}$ corresponding to the set of pseudonyms, where $\varsigma_A \in Z_p^*$ is the master secret selected by TA for A. Note that every user can query TA for public/private key pairs of its pseudonym set.

4.1.2 Revised Structured Encryption

The structured encryption proposed by Chase and Kamara[34] solves the problem of storing a set of data with specific structure on untrusted storage. The data structure introduced in their scheme could be matrix, labeled data and graph. Here, we modify their scheme in two orthogonal directions to fulfill our design requirements. First, original structured encryption scheme relies on the symmetric encryption, which lacks the scalability in the public key cryptosystem. Second, we endow more functionalities to central storage (CS), such as verification, homomorphic operation, and proxy re-encryption. Without loss of generality, we first give a brief review of the modified labeled data scheme that we use in our system in the subsequent development.

Define two pseudo random functions $\mathcal{F}$ and $\mathcal{G}$, where $\mathcal{F} : \{0,1\}^k \times \mathcal{A} \to \{0,1\}^{\max(L)\log n}$ and $\mathcal{G} : \{0,1\}^k \times \mathcal{A} \to \{0,1\}^k$. For each attribute of users $x \in \mathcal{A}$, we define the set $L(x) = \{i \in [n] | (i, \sigma) \in L\}$ as the label of the corresponding attribute, where $x \in Z_p$ represents the attribute and $n \in \mathbb{N}$ is a sufficiently large natural number. The modified labeled data encryption scheme $\mathsf{MLabel} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Token}, \mathsf{Search}, \mathsf{Retrieval}, \mathsf{Dec})$ is de-

fined as follows, where the sub-protocols $\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}$ include the selective-tag encryption scheme:

$\mathsf{Gen}(1^{\xi})$: input the security parameter ξ and output two k-bit keys K_1 and K_2 and the user's public key $\overline{pk}$.

$\mathsf{Enc}(K_1, K_2, \overline{pk}, L, \sigma)$: construct a searchable dictionary $\mathcal{T}$ in CS:

1) The user defines a random permutation $\mathbb{P} : [n] \to [n]$. For each attribute $x \in \mathcal{A}$, let $kw := \mathcal{G}_{K_2}(x)$ and store $\mathbb{P}(i)_{i \in L(x)} \oplus \mathcal{F}_{K_1}(x)$ in $\mathcal{T}$ together with kw.

2) Let w_i, $1 \leqslant i \leqslant n$ represent a sequence of data items which consist of multiple copies of σ^{-1} and a certain number of dummy strings. Apply the permutation $\mathbb{P}$ for the sequence of data items to generate w_j^*.

3) Use selective-tag encryption scheme[35] to encrypt the w_j^* as $c_j = \mathcal{E}_{\overline{pk}}(w_j^*)$ with different random numbers, where j denotes the index of each ciphertext.

4) Store the dictionary $\mathcal{T}$ together with the ciphertext c_j in CS.

$\mathsf{Token}(K_1, K_2, x)$: a user generates the token $\tau := (\mathcal{F}_{K_1}(x), \mathcal{G}_{K_2}(x))$ if he/she wants to issue to the voter for reputation voting on the attribute x.

$\mathsf{Search}(\mathcal{T}, \tau)$: when CS responds to the subprotocol Search, it first computes $\mathcal{T}(\mathcal{G}_{K_2}(x) \oplus \mathcal{F}_{K_1}(x))$ and outputs $\mathcal{J} := (j_1, j_2, ..., j_j)$, then it may perform operation on the ciphertext set c_j which the corresponding entries that $\mathcal{J}$ points to.

$\mathsf{Retrieval}(\mathcal{T}, \mathcal{G}_{K_2}(x), \overline{pk}_{i \to i'})$: the central storage performs the proxy re-encryption using $\overline{pk}_{i \to i'}$ on the entries which have been operated before. Queriers may retrieve the reputation value by searching for $\mathcal{T}(\mathcal{G}_{K_2}(x))$.

$\mathsf{Dec}(c_j, \overline{sk}_{i'})$: queriers who run the $\mathsf{Retrieval}$ subprotocol are able to use their private keys $\overline{sk}_{i'}$ to decrypt and collect the designated vote receiver's reputation value.

4.1.3 Review of Non-Interactive Proof System

We implement the non-interactive proof system proposed by Groth and Sahai[19], which is an efficient system for bilinear groups. However, their work is based on general cases for bilinear pairing without considering the scenarios in our proposed system. In what follows, we give a brief introduction to their proposed system.

Setup. As in previous section, assume that we have a bilinear map $e : G_1 \times G_1 \to G_2$. With entry-wise multiplication, we can get the Z_p-module $M_1 = G_1^3$, in which we define another bilinear map $\hat{e} : M_1 \times M_1 \to M_2$. Note our system is based on the decision linear assumption introduced by Boneh *et al.* in [36] stating that given three random generators $f, h, g \in G_1$ and f^r, h^s, g^t, it is hard to distinguish the case $t = r + s$ from t random. In the main design of our system, we use the module $M_2 = G_2^6$ given by entry-wise multiplication. The symmetric bilinear map $\hat{e}_6 : G_1^3 \times G_1^3 \to G_2^6$ is given by

$$\hat{e}_6\left(\begin{pmatrix} a \\ b \\ c \end{pmatrix}, \begin{pmatrix} x \\ y \\ z \end{pmatrix}\right) \to \begin{pmatrix} e(a,x) & e(a,y)e(b,x) & e(a,z)e(c,x) \\ 0 & e(b,y) & e(b,z)e(c,y) \\ 0 & 0 & e(c,z) \end{pmatrix}.$$

Lemma 1[19]. *Define a map $\mu : Z_p^9 \to M_2$, $\forall u_1, u_2, u_3 \in M_1$, $\exists \rho_{11}, \rho_{12}, ..., \rho_{33} \in Z_p^9$ and $t_1, ..., t_H \in Z_p$, such that $\prod_{i=1}^{3}\prod_{j=1}^{3} \hat{e}_6(u_i, u_j)^{\rho_{ij}} = 1$, where $\rho_{ij} = \sum_{h=1}^{H} t_h \eta_{hij}$ and $\eta_{hij} \in Z_p$.*

For perfect soundness of the NIWI and NIZK proof, the common reference string and simulated reference strings must be computationally indistinguishable, and thus we also have $\mu(\eta_h) = 1$ for all $\eta_h \in Z_p^9$ and $\eta_{hij} \in \eta_h$ performs as the basis for generating the kernel of μ.

CRS Generation. TA also publishes the common reference string (crs) on the bulletin board for every user to verify the proofs. The above generation process outputs a set of parameters (p, G_1, G_2, e, g). TA randomly picks $\alpha, \beta, r_u, s_v \leftarrow Z_p^*$. Set $f = g^{\alpha}, h = g^{\beta}$, and generate $u_i \in M_1$, which are $u_1 := (f, 1, g), u_2 := (1, h, g)$ and $u_3 := (f^{r_u}, h^{s_v}, g^{t_w})$, where $t_w = r_u + s_v$. TA sets the crs as $(p, G_1, G_2, e, M_1, M_2, \hat{e}, g, u_1, u_2, u_3, \eta_1, \eta_2, \eta_3)$ and publishes it on the bulletin board.

Proof Generation. We will use NIWI and NIZK together or separately based on different steps in our proposed system. Note that NIWI proof tries to convince that the witnesses in the statement are indistinguishable, where the verifier or adversaries cannot locate which witness (prover) corresponds to the statement, while NIZK proof represents that the statement can be verified without exposing any other information. We denote the credential of A's unique attribute as $x_q^A \in G_1$, while there could be other variables in the bilinear paring, i.e., $y_q \in G_1$, where q is the index of different variables or credentials. Suppose x_q and y_q can form an equation $\prod_{q=1}^{Q} e(x_q, y_q) = \mathbb{T}$, where $\mathbb{T} \in M_2$. Given the elements from M_1 and random numbers which form the commitments of both x_q and y_q, users can make the corresponding NIWI or NIZK proof π based on two commitments $Com(x_q)$ and $Com(y_q)$. Note that certificates of the same attribute value on different real identities

are the same. In our scheme, users are given the authority to generate unique proofs and commitments on their verified attribute and the corresponding value for verification.

Verification. Trust authority has a bulletin board for publishing the common reference string (crs) used for every member in the system to verify the given proofs. Given proof π_i, $Com(x_q)$, $Com(y_q)$, public parameter u_i and the statement $\prod_{q=1}^{Q} \hat{e}(x_q, y_q) = \mathbb{T}$ that we are concerned, we can set up the following verification equation for a verifier to publicly verify the corresponding statements or equations:

$$\prod_{q=1}^{Q} e_6(Com(x_q), Com(y_q)) = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & \mathbb{T} \end{pmatrix} \prod_{i=1}^{3} \hat{e}_6(u_i, \pi_i). \quad (1)$$

If the check passes, we can learn that the variables x_q and y_q satisfy the statement while ensuring that the verifier learns nothing about x_q and y_q.

4.2 Attribute Verification

In this subsection, we present the attribute verification process between a voter and a vote receiver. The design purpose of attribute verification is to request a vote receiver to grant the vote privilege to a voter. During the protocol run, we need to guarantee the *voter anonymity* which allows a voter to use his/her pseudonyms to hide his/her real identity and obtain the privilege based on the verification result.

4.2.1 Credential Issuance

Trust authority picks a random integer $\tilde{x}_\imath \in Z_p^*$ for a specific class of attribute, where $\imath$ could represent the attribute that the user allows TA to verify, such as age, occupation, and gender. Once it successfully verifies the specific attribute, it computes the credentials $\tilde{v}_\imath = g^{\tilde{x}_\imath} \in G_1$ and $e(g, g) \in G_2$. The public key tuple for the verification is $(g, \tilde{v}_\imath, e(g, g))$, while the secret/sign key held by TA is $\tilde{x}_\imath$ corresponding to different attributes. Intuitively, users can show their validity by presenting $\tilde{v}_\imath$. However, we cannot directly reveal the credentials $\tilde{v}$ to the public verifiers, which may incur impersonation attack when adversaries use the credentials to show their validity on specific attributes.

As an illustration, to prove user A's occupation which has been verified, we implement the certified signature scheme in [21, 37] for the verification of valid $\tilde{v}$. TA randomly picks group elements $\hat{f}, \hat{h}, \hat{z} \in G_1$, and publishes the tuple $(\hat{f}, \hat{h}, \Gamma)$ as the authority key to verify $\tilde{v}$, where $\Gamma = e(\hat{f}, \hat{z}) \in G_2$ and $\hat{z}$ is a secret key for TA. Then, TA picks a random number $\hat{r} \in Z_p$ and computes $(a, b) := (\hat{f}^{-\hat{r}}, (\hat{h}\tilde{v})^{\hat{r}}\hat{z})$. Given $\tilde{v}$, everyone is able to verify the valid credential by checking $e(a, \hat{h}\tilde{v})e(\hat{f}, b) = \Gamma$.

4.2.2 Proof Generation and Verification

Here, we apply the NIWI proof for the verification of users' credential on a specific attribute. For *voter anonymity*, we require that the vote receiver does not distinguish different users or trace a specific user based on the verification information that the voter shows. In this case, the voter V wants to vote on the message published by vote receiver R. Note that the message is associated with one of R's verified attributes, e.g., *occupation*. Also, V needs to prove to R that he/she has the same verified attribute. However, in this stage, we do not require that V's and R's attribute values are identical, since the distinctness does not affect the voting process.

1) *Proof Generation.* As we can see in the check equation $e(a, \hat{h}\tilde{v})e(\hat{f}, b) = \Gamma$, there are only two variables that we want to hide, $\tilde{v}$ and b. The original schemes[21,37] are not concerned about revealing the credential $\tilde{v}$, while we need to keep those checking processes continuing without exposing the plaintext value of $\tilde{v}$. Therefore, we need to commit those two variables instead of all the parameters in the bilinear map. In this case, the voter V uses the parameters from the published bulletin board messages $u_1, u_2, u_3 \in M_1$ and chooses $r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23} \in Z_p^*$ to commit $\tilde{v}$ and b as follows, $Com(\hat{h}\tilde{v}) := \overline{c}_0 := (1, 1, \hat{h}\tilde{v})u_1^{r_{11}}u_2^{r_{12}}u_3^{r_{13}}$ and $Com(b) := \overline{d}_0 := (1, 1, (\hat{h}\tilde{v})^{\hat{r}}\hat{z})u_1^{r_{21}}u_2^{r_{22}}u_3^{r_{23}}$. Apart from this, V also generates a set of NIWI proofs,

$$\pi_i := (1, 1, \hat{f}^{-\hat{r}})^{r_{1i}}(1, 1, \hat{f})^{r_{2i}}.$$

Then, V sends the packet $\mathsf{P} := (\overline{c}_0, \overline{d}_0, \pi_1, \pi_2, \pi_3)$ to the public domain for verification.

2) *Public Verification.* For *voter anonymity*, user V may not want to expose his/her real identity or the credentials to the third party or the public domain. As a result, given the public parameters crs and $(\hat{f}, \hat{h}, \Gamma)$, vote receiver R can verify the validity of the corresponding credentials. Similar to the bilinear map $\hat{e}_6 : G_1^3 \times G_1^3 \to G_2^6$, we define another bilinear map

$\hat{e}_9 : G_1^3 \times G_1^3 \to G_2^9$:

$$\hat{e}_9\left(\begin{pmatrix}a\\b\\c\end{pmatrix},\begin{pmatrix}x\\y\\z\end{pmatrix}\right) \to \begin{pmatrix}e(a,x) & e(a,y) & e(a,z)\\ e(b,x) & e(b,y) & e(b,z)\\ e(c,x) & e(c,y) & e(c,z)\end{pmatrix}.$$

R verifies the validity of the corresponding credential by checking the equality of the following equation,

$$\begin{aligned}&\hat{e}_9\left(\overline{c}_0,\begin{pmatrix}1\\1\\\hat{f}^{-\hat{r}}\end{pmatrix}\right)\cdot\hat{e}_9\left(\overline{d}_0,\begin{pmatrix}1\\1\\\hat{f}\end{pmatrix}\right)\\ \stackrel{?}{=}\ &\begin{pmatrix}1&1&1\\1&1&1\\1&1&\Gamma\end{pmatrix}\prod_{i=1}^{3}\hat{e}_9(u_i,\pi_i).\end{aligned}$$

Lemma 2[19]. *Let M_1, M_2 be Z_p-modules, for all $r \in Z_p$, $u, u', v \in M_1$, we have $\hat{e}(u^r u', v) = \hat{e}(u, v)^r \hat{e}(u', v)$.*

Accordingly, the left hand side of the above check equation can be derived as follows,

$$\begin{aligned}LHS \ =\ &\hat{e}_9\left(\begin{pmatrix}f^{r_{11}+r_u r_{13}}\\h^{r_{12}+s_v r_{13}}\\\hat{h}\tilde{v}\end{pmatrix},\begin{pmatrix}1\\1\\\hat{f}^{-\hat{r}}\end{pmatrix}\right)\\ &\hat{e}_9\left(\begin{pmatrix}f^{r_{11}+r_u r_{13}}\\h^{r_{12}+s_v r_{13}}\\g^{\dot{r}}\end{pmatrix},\begin{pmatrix}1\\1\\\hat{f}^{-\hat{r}}\end{pmatrix}\right)\\ &\hat{e}_9\left(\begin{pmatrix}f^{r_{21}+r_u r_{23}}\\h^{r_{22}+s_v r_{23}}\\(\hat{h}\tilde{v})^{\hat{r}}\hat{z}\end{pmatrix},\begin{pmatrix}1\\1\\\hat{f}\end{pmatrix}\right)\\ &\hat{e}_9\left(\begin{pmatrix}f^{r_{21}+r_u r_{23}}\\h^{r_{22}+s_v r_{23}}\\g^{\ddot{r}}\end{pmatrix},\begin{pmatrix}1\\1\\\hat{f}\end{pmatrix}\right).\end{aligned}$$

For an illustration, we only show the verification of the intersection of row 3 and column 3 of the corresponding matrix,

$$\begin{aligned}&LHS_{33}\\ =\ &e(\hat{h}\tilde{v}, \hat{f}^{-\hat{r}})e(g^{\dot{r}}, \hat{f}^{-\hat{r}})e((\hat{h}\tilde{v})^{\hat{r}}, \hat{f})e(\hat{z}, \hat{f})e(g^{\dot{r}}, \hat{f})\\ =\ &e(\hat{z}, \hat{f})e(g, f)^{\ddot{r}-\hat{r}\dot{r}},\end{aligned}$$

where $\dot{r} = r_{11} + r_{12} + t_w r_{13}$ and $\ddot{r} = r_{21} + r_{22} + t_w r_{23}$. Meanwhile, the right hand side of the equation can be derived as follows,

$$\begin{aligned}RHS_{33} \ =\ &\Gamma e(g, \hat{f}^{-\hat{r}r_{11}+r_{21}})e(g, \hat{f}^{-\hat{r}r_{12}+r_{22}})\\ &e(g, \hat{f}^{-\hat{r}r_{13}+r_{23}})\\ =\ &\Gamma e(g, \hat{f})^{-\hat{r}\dot{r}+\ddot{r}}.\end{aligned}$$

It is obvious that if the above two equations are equal, the voter has proved that his/her attribute has been verified by TA. Meanwhile, the vote receiver learns nothing about the credential and the voter's identity.

4.2.3 Voting Entry Issuance

As far as the vote receiver R verifies voter V's attribute, R can issue V the voting privilege which includes the voting entry. The voting entries are indexed by the keyword $kw := \mathcal{G}_{K_2}(x)$, so that we cannot allow any voter to obtain the information, or malicious users may arbitrarily tamper the values in CS. To avoid inconsistency, we hereby list the attribute verification process as a whole,

1) voter $V \to$ vote receiver R: $\overline{c}_0, \overline{d}_0, \pi_1, \pi_2, \pi_3$;
2) $R \to V : E_{pk_{CS}}(\mathcal{F}_{K_1}(x), \mathcal{G}_{K_2}(x)), \tilde{t}_1$,
   $U = \hat{H}(vk_R||\mathcal{M}||L(x)||seq)$,
   $SIG(E_{pk_{CS}}(\mathcal{F}_{K_1}(x), \mathcal{G}_{K_2}(x))||\tilde{t}_1||U)$,

where pk_{CS} is the public key of CS and vk_R is the vote key of the vote receiver which we will discuss in later subsection. The vote receiver computes $U = \hat{H}(vk_R||\mathcal{M}||L(x)||seq)$ for the voter, where $\mathcal{M} \in Z_p^*$ together with x is the attribute associated message and $\hat{H} : \{0,1\}^* \to Z_p^*$ is a collision-resistant hash function. We also use a sequence number seq to uniquely identify the U, where seq is determined by the vote receiver to avoid the voter's "double voting" and it always changes every time when there is a query. We refer $\tilde{t}_1$ as the timestamp to avoid reply attack. After obtaining the encrypted voting entry from the vote receiver, V is able to perform attribute-based reputation voting.

4.3 Attribute-Based Reputation Voting Privilege Check

This protocol is mostly run by a voter and CS. To guarantee no voter "double votes" the message, we have a voting privilege check before the anonymous voting protocol. Specifically speaking, the "double voting" happens when a voter wants to use his/her attribute and the corresponding value to vote the message that he/she has already voted. To take a step further, we allow the voter to use the same attribute and the corresponding value to vote different messages associated with the same attribute. For example, it is permitted to use the same voter's attribute, *occupation: mechanic*, to vote on the different attribute-associated messages $\mathcal{M}_1$: *Firestone is a good car shop* and $\mathcal{M}_2$: *Midas is not as good as Firestone on tire service.* However, if one voter or even the malicious vote receiver tries to vote on the same attribute associated message, they can be

easily identified in our scheme. For the ease of description, we list the whole process before the voting, some of which is drawn from the existing e-cash scheme in [38].

Without loss of generality, we review and modify the compact e-cash scheme to fit our proposed system. Note that a voter in our system uses assigned pseudonym for voting to avoid identity leakage during the protocol run.

KeyGen(1^ξ): TA generates a signature key pair $(pks_{\rm TA}, sks_{\rm TA})$ to sign the message M such that $Z_p \times Z_p \times Z_p \subseteq \mathsf{M}$. Each valid user is given a unique vote key pair $(vk_{\rm ID}, vsk_{\rm ID}) = (g^\phi, \phi)$ for a randomly selected $\phi \in Z_p$ by TA.

Authorization(Voter$(pks_{\rm TA}, vsk_V)$, TA$(vk_V, sks_{\rm TA})$): first, the voter needs to prove to TA the knowledge of vsk_V by using $PK\{(\phi) : y = g^\phi\}$. Then, both TA and V need to contribute to the voter's secret ε, and V also selects ϖ as another secret. The process is as follows, V selects random values $\varepsilon', \varpi \in Z_p$ and sends a commitment $\mathbb{A}' = \hat{h}^{r'}\hat{g}_1^{\phi}\hat{g}_2^{\varepsilon'}\hat{g}_3^{\varpi}$ to TA. TA then sends a random $\varepsilon'' \in Z_p$ back to V. Then, V sets $\varepsilon = \varepsilon' + \varepsilon''$. Thus, TA and V can locally compute the commitment $\mathbb{A} = \hat{g}_2^{\varepsilon'}\mathbb{A}' = \hat{h}^{r'}\hat{g}_1^{\phi}\hat{g}_2^{\varepsilon}\hat{g}_3^{\varpi}$. Using the signature scheme in [39], the voter is able to obtain the signature $\sigma_{\rm TA}$ on committed values $(vsk_V, \varepsilon, \varpi)$ contained in $\mathbb{A}$.

CertIssue(Voter$(m_{\imath.\jmath})$, TA$(\tilde{x}_\imath)$): TA uses $\tilde{x}_\imath$ to sign each specific value of the corresponding attribute after the verification of a particular attribute value. We assume the value of an attribute in our system could be represented as $m_{\imath.\jmath} \in Z_p$, where $\imath$ denotes the general classification of attributes and $\jmath$ is the specific value of those attributes. Note that this subprotocol runs before the voting privilege check process. When the protocol ends, TA outputs the certificate as $\sigma_{\imath.\jmath} = g^{1/(\tilde{x}_\imath + m_{\imath.\jmath})} \in G_1$.

VotePrivGen (Voter $(vsk_V, \varepsilon, \varpi, \sigma_{\rm TA}, \sigma_{\imath.\jmath}, U)$, $\mathcal{M}$, crs): voter V generates a vote serial number $S = g^{1/\hat{H}(\sigma_{\imath.\jmath})+\mathcal{M}+\varepsilon+1} \in G_1$ and a double-voting equation $T = vk_V(g^{1/\hat{H}(\sigma_{\imath.\jmath})+\mathcal{M}+\varpi+1})^U \in G_1$. Then, V chooses random numbers $\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5 \in Z_p$ and computes a set of commitments as $\mathbb{B} = \hat{h}^{\lambda_1}\hat{g}_1^{\phi}$, $\mathbb{C} = \hat{h}^{\lambda_2}\hat{g}_1^{\varepsilon}$, $\mathbb{D} = \hat{h}^{\lambda_3}\hat{g}_1^{\varpi}$, $\mathbb{E} = \hat{h}^{\lambda_4}\hat{g}_1^{\hat{H}(\sigma_{\imath.\jmath})}$ and $\mathbb{F} = \hat{h}^{\lambda_5}\hat{g}_1^{\mathcal{M}}$; and it proves to CS the knowledge of $\sigma_{\rm TA}$ on openings of $\mathbb{B}$, $\mathbb{C}$ and $\mathbb{D}$. Also, the voter V needs to prove he/she has the knowledge on the serial number S and T. Finally the above proof is the following proof of knowledge, $PK\{(\vartheta, \varrho, \delta, \gamma_1, \gamma_2, \gamma_3) : \hat{g}_1 = (\mathbb{FEC})^\vartheta\hat{h}^{\gamma_1} \wedge S = g^\vartheta \wedge \hat{g}_1 = (\mathbb{FED})^\varrho\hat{h}^{\gamma_2} \wedge \mathbb{B} = \hat{g}_1^\delta\hat{h}^{\gamma_3} \wedge T = g^\delta(g^U)^\varrho\}$. To this end, voter V uses Fiat-Shamir heuristic[40] to turn all the proofs above into one signature of knowledge on the values $\Sigma := (S, T, \mathbb{B}, \mathbb{C}, \mathbb{D}, \mathbb{E}, \mathbb{F}, \hat{g}_1, \hat{h}, \hat{n}, g, U)$ into a resulting signature Ψ, which is used to trace back the malicious user (refer to Subsection 5.1).

VotePrivCheck(Voter(Σ, Ψ), CS): voter V uses the proofs above and the corresponding signature to prove the validity of the vote. However, we separate the vote privilege check process and the anonymous voting scheme for the ease of description. For real implementation, we would rather combine the NIWI and NIZK proofs introduced in Subsection 4.4 into Σ and Ψ for consistency and authenticity of the vote. In this subprotocol, if the Ψ and all proofs check pass, CS accepts the following attribute-based vote based on the fact that "double-voting" has also been checked. Otherwise, given the existing voting-equation and the corresponding Σ and Ψ, a malicious voter who "double votes" the message $\mathcal{M}$ using the associated attribute label $L(x_i)$ can be easily identified using the following equation:

$$\begin{aligned}
&\left(\frac{T_2^{U_1}}{T_1^{U_2}}\right)^{(U_1-U_2)^{-1}} \\
=\ &\left(\frac{g^{\phi U_1 + \frac{U_1U_2}{(\hat{H}(\sigma_{\imath.\jmath})+\mathcal{M}+\varpi+1)}}}{g^{\phi U_2 + \frac{U_1U_2}{(\hat{H}(\sigma_{\imath.\jmath})+\mathcal{M}+\varpi+1)}}}\right)^{(U_1-U_2)^{-1}} \\
=\ &g^{\frac{\phi(U_1-U_2)}{U_1-U_2}} = g^\phi = vk_V,
\end{aligned}$$

where (T_2, U_2) is referred to as the voting privilege that the malicious voter wants to "double vote" a particular attribute-associated message. According to the previous assumption, we do not consider the scenario where V uses $\mathcal{M}_2$ and (T_2, U_2) to pass the checking process, and uses the encrypted attribute to vote his/her opinion on $\mathcal{M}_1$. Thus, both (T_1, U_1) and (T_2, U_2) yield the same serial number S, which helps central storage locate the record of voting history in order to check the "double-voting". Since the vote key vk uniquely identifies the user in the system, CS checks all the votes indexed by the serial number S in its database to identify and report the designated "double-voter" no matter which pseudonym he/she uses.

4.4 Attribute-Based Anonymous Voting

In this stage, voters use their verified attributes and the corresponding values to vote on the attribute-associated message $\mathcal{M}$, while CS is responsible for collecting the votes.

4.4.1 Attribute Value Check

Similar to the attribute verification process in Subsection 4.2, we also need verification on that attribute value, such that CS can be ensured the attribute value for voting has already been verified by TA. We also apply the NIWI proof for verifying attribute certificate $\sigma_{\imath.\jmath}$.

Committing to Variables. We need to give NIWI proofs for the following equations. We also list the validity verification equation in Subsection 4.2 to denote that both equations must be simultaneously satisfied and verified by CS:

$$e(a, \hat{h}\tilde{v}_\imath)e(\hat{f}, b) = \Gamma, \tag{2}$$

$$e(\sigma_{\imath.\jmath}, \tilde{v}_\imath g^{m_{\imath.\jmath}})e(g^{-1}, g) = 1. \tag{3}$$

Voter V chooses $u_i \in M_1$ from the published crs to commit those variables as follows. Taking (3) as an example, V first chooses elements $u_1, u_2, u_3 \in M_1$ from crs and randomly selects $r_1, r_2, r_3, r'_1, r'_2, r'_3 \leftarrow Z_p$. Then, V computes

$$\begin{aligned} Com(\sigma_{\imath.\jmath}) &:= \overline{c}_1 = (1, 1, \sigma_{\imath.\jmath})u_i^{r_i} \\ &= (f^{r_1+r_3r_u}, h^{r_2+r_3s_v}, \sigma_{\imath.\jmath}g^{r_1+r_2+r_3(r_u+s_v)}), \end{aligned}$$

and V also commits to the other variable as

$$\begin{aligned} Com(\tilde{v}_\imath g^{m_{\imath.\jmath}}) &:= \overline{d}_1 = (1, 1, \tilde{v}_\imath g^{m_{\imath.\jmath}})u_i^{r'_i} \\ &= (f^{r'_1+r'_3r_u}, h^{r'_2+r'_3s_v}, \\ &\quad \tilde{v}_\imath g^{m_{\imath.\jmath}+r'_1+r'_2+r'_3(r_u+s_v)}). \end{aligned}$$

Generating NIWI *Proof.* For voter V, he/she needs to generate the NIWI proof $\dot{\pi}_i$ by using the parameters in crs and those in the commitments. Given the kernel vectors of μ_6 in crs, $\boldsymbol{\eta}_1 := (0, 1, 0, -1, 0, 0, 0, 0, 0)$, $\boldsymbol{\eta}_2 := (0, 0, 1, 0, 0, 0, -1, 0, 0)$, $\boldsymbol{\eta}_3 := (0, 0, 0, 0, 0, 1, 0, -1, 0)$, user A randomly selects $t_1, t_2, t_3 \in Z_p$ to generate the NIWI proofs as follows:

$$\dot{\pi}_i = \prod_{j=1}^{3} u_j^{\sum_{h=1}^{3} t_h \eta_{hij}} (1, 1, \sigma_{\imath.\jmath})^{r'_i} \overline{d}_1^{r_i}.$$

Verification. The voter sends the packet $\mathsf{P}' := (\overline{c}_1, \overline{d}_1, \dot{\pi}_1, \dot{\pi}_2, \dot{\pi}_3)$ to CS for the verification. We can modify (3) to $e(\sigma_{\imath.\jmath}, \tilde{v}_\imath g^{m_{\imath.\jmath}}) = e(g, g)$, where $e(g, g)$ is a known factor. Based on the crs, CS can check (3) according to (4) as what follows:

$$LHS = \hat{e}_6\left(\begin{pmatrix} f^{r_1+r_3r_u} \\ h^{r_2+r_3s_v} \\ \sigma_{\imath.\jmath}g^{\hbar} \end{pmatrix}, \begin{pmatrix} f^{r'_1+r'_3r_u} \\ h^{r'_2+r'_3s_v} \\ \tilde{v}_\imath g^{m_{\imath.\jmath}+\hbar'} \end{pmatrix}\right), \tag{4}$$

where $\hbar = r_1 + r_2 + r_3(r_u + s_v)$ and $\hbar' = r'_1 + r'_2 + r'_3(r_u + s_v)$, respectively. The central storage then checks the right hand side as follows,

$$\begin{aligned} &\prod_{i=1}^{3} \hat{e}_6(u_i, \dot{\pi}_i) \\ =\ &\prod_{i=1}^{3} \hat{e}_6(u_i, (1, 1, \sigma_{\imath.\jmath})^{r'_i} \overline{d}_1^{r_i}) \\ =\ &\hat{e}_6\left(\prod_{i=1}^{3} u_i^{r'_i}, (1, 1, \sigma_{\imath.\jmath})\right) \hat{e}_6\left(\prod_{i=1}^{3} u_i^{r_i}, \overline{d}_1\right) \\ =\ &\hat{e}_6\left(\begin{pmatrix} f^{r'_1+r'_3r_u} \\ h^{r'_2+r'_3s_v} \\ g^{\hbar'} \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ \sigma_{\imath.\jmath} \end{pmatrix}\right) \cdot \\ &\hat{e}_6\left(\begin{pmatrix} f^{r_1+r_3r_u} \\ h^{r_2+r_3s_v} \\ g^{\hbar} \end{pmatrix}, \begin{pmatrix} f^{r'_1+r'_3r_u} \\ h^{r'_2+r'_3s_v} \\ \tilde{v}_\imath g^{m_{\imath.\jmath}+\hbar'} \end{pmatrix}\right). \end{aligned}$$

Note that we omit several terms in the proof $\dot{\pi}_i$ since we have $\prod_{i=1}^{3}\prod_{j=1}^{3} \hat{e}_6(u_i, u_j)^{\rho_{ij}} = 1$ according to Lemma 1. By directly applying the bilinear operation to the above equations, we can easily check that all the entries satisfy the equalities in (4) except the last one in the matrix. However, we observe the result in the row 3 and column 3 of the corresponding matrix. We expand all the pairing results as follows,

$$\begin{aligned} LHS_{33} &= e(\sigma_{\imath.\jmath}g^{\hbar}, \tilde{v}_\imath g^{m_{\imath.\jmath}+\hbar'}) \\ &= e(g, g)^{\left(\frac{1}{\tilde{x}_\imath + m_{\imath.\jmath}} + \hbar\right)(\tilde{x}_\imath + m_{\imath.\jmath} + \hbar')} \\ &= e(g, g)^{1+\hbar(\tilde{x}_\imath + m_{\imath.\jmath}) + \frac{\hbar'}{\tilde{x}_\imath + m_{\imath.\jmath}} + \hbar\hbar'}, \end{aligned}$$

and according to (1),

$$\begin{aligned} RHS_{33} &= Te(g^{\hbar'}, \sigma_{\imath.\jmath})e(g^{\hbar}, \tilde{v}_\imath g^{m_{\imath.\jmath}+\hbar'}) \\ &= e(g, g)^{1+\frac{\hbar'}{\tilde{x}_\imath + m_{\imath.\jmath}} + \hbar(\tilde{x}_\imath + m_{\imath.\jmath} + \hbar')}, \end{aligned}$$

and two results are identical.

Until now, CS is convinced that the voter has the voting attribute and the corresponding value which is verified by TA. Note that we update the existing Σ as $\Sigma' := (S, T, \mathbb{B}, \mathbb{C}, \mathbb{D}, \mathbb{E}, \mathbb{F}, \hat{g}, \hat{h}, \hat{n}, g, U, \mathsf{P}, \mathsf{P}')$, and generate a signature Ψ' for CS. If all the verification checks pass, we can allow the user to perform the anonymous voting. Otherwise, CS aborts the protocol and records the log files.

4.4.2 *Equality Check for* NIWI *Proof*

The design requirement for the anonymous voting is two folds: first, we need to guarantee the encrypted attribute value in the vote should be consistent with the one used in the previous NIWI proofs; second, the vote itself should not reveal any information related to the real identity of the voter.

For each voter, we recall the subprotocol Search which gives a voter the direct voting entry to vote. A voter sends $E_{pk_{CS}}(\mathcal{F}_{K_1}(x), \mathcal{G}_{K_2}(x)), \tilde{t}_1$ and U together with the signature to CS until it outputs the pointer $\mathcal{J} = j$ which points to a specific ciphertext w_j^*. According to Subsection 4.1.2, we let $\overline{pk}$ denote the public key used to encrypt the attribute value. Indeed, we apply the selective-tag encryption[35] as our basic encryption scheme, since it has the encrypted form which perfectly matches the commitment scheme implemented in the NIWI proof. We briefly review the selective-tag encryption scheme.

Key Generation (STKeyGen). TA assigns a tuple $\overline{pk} := (\overline{f}, \overline{h}, \overline{k}, \overline{l}) \in G_1^4$ as a public key for a user, and distributes the private key (χ, ψ) to the user where $\overline{f} = g^{\chi}, \overline{h} = g^{\psi}$.

Encryption (STEnc). The user chooses random numbers $\overline{r}, \overline{s} \in Z_p$ and selects a public tag $\overline{t}$ to encrypt a message $\overline{m}$ as $\mathcal{E}(\overline{m}) := (\overline{f}^{\overline{r}}, \overline{h}^{\overline{s}}, g^{\overline{r}+\overline{s}}\overline{m}, (g^{\overline{t}}\overline{k})^{\overline{r}}, (g^{\overline{t}}\overline{l})^{\overline{s}})$.

Decryption (STDec). Decryption can be done by computing $\overline{m} = g^{\overline{r}+\overline{s}}\overline{m}(\overline{f}^{\overline{r}})^{-1/\chi}(\overline{h}^{\overline{s}})^{-1/\psi}$.

Then, we show the generation process of NIZK proofs which prove the equality of the content in the commitment and the corresponding encrypted attribute value.

1) NIZK *Proof Generation.* The generation process of NIZK is similar to NIWI in [21], where we commit to the exponential of the generator $g \in G_1$. We first show the statements that we need to prove. Given the ciphertext of an encrypted certificate as $\mathcal{E}(\sigma_{\imath.\jmath}) := (X_1, X_2, X_3, X_4, X_5) = (f^{\overline{r}_e}, h^{\overline{s}_e}, g^{\overline{r}_e+\overline{s}_e}\sigma_{\imath.\jmath}, (g^{\overline{t}}\overline{k})^{\overline{r}_e}, (g^{\overline{t}}\overline{l})^{\overline{s}_e})$. According to the previous subsections, $Com(\sigma_{\imath.\jmath}) := (C_1, C_2, C_3) = (f^{r_1+r_3r_u}, h^{r_2+r_3s_v}, \sigma_{\imath.\jmath}g^{r_1+r_2+r_3(r_u+s_v)})$. Setting $r_0 = \overline{r}_e - r_1$ and $s_0 = \overline{s}_e - r_2$, we have the following statements that need to be simultaneously satisfied for proving the equality of the committed value and the encrypted value,

$$\begin{aligned}&\varphi = 1 \wedge (C_1^{-1}X_1)^{\varphi} f^{r_0}(f^{r_u})^{r_3} = 1 \wedge\\ &(C_2^{-1}X_2)^{\varphi} f^{s_0}(f^{s_v})^{r_3} = 1 \wedge\\ &(C_3^{-1}X_3)^{\varphi} g^{r_0+s_0}(f^{r_u+s_v})^{r_3} = 1,\end{aligned} \tag{5}$$

where we cannot disclose φ, r_0, s_0, r_3 to the verifier and we need to prove the above equations given the ciphertext and the commitments. Note that $\wedge$ denotes "and".

To generate the commitments, we randomly select two numbers $\theta, \zeta \in Z_p$ and add one more universal parameter $u := u_1^{\theta}u_2^{\zeta}$ onto crs, which is linearly independent of u_1 and u_2. Taking the second equation in (5) as an example, we need to commit to the exponentials φ, r_0, r_3 as $\mathcal{C}_1 = u^{\varphi}u_1^{\nu_1}u_2^{\nu_2}$, $\mathcal{C}_2 = u^{r_0}u_1^{\nu_1}u_2^{\nu_2}$ and $\mathcal{C}_3 = u^{r_3}u_1^{\nu_1}u_2^{\nu_2}$, where $\nu_i \in Z_p$ are random numbers. Based on the commitments, user A can generate NIZK proof as follows,

$$\ddot{\pi}_i = (1, 1, C_1^{-1}X_1)^{\nu_i}(1, 1, f)^{\nu_i}(1, 1, f^{r_u})^{\nu_i}.$$

Since V has already sent the commitment of $\sigma_{\imath.\jmath}$ to CS in P', CS is able to compute $C_i^{-1}X_i$ after V delivers the ciphertexts.

2) *Verification.* We do not give the detail of the checking process, because the process is similar to (2),

$$\prod_{i=1}^{3}\hat{e}_9\left(\begin{pmatrix}1\\1\\Y_i\end{pmatrix}, \mathcal{C}_i\right) = \hat{e}_9\left(\begin{pmatrix}1\\1\\1\end{pmatrix}, u\right)\prod_{i=1}^{2}\hat{e}_9(\ddot{\pi}_i, u_i),$$

where Y_i is the elements in the set $(C_1^{-1}X_1, f, f^{r_u})$, respectively. The central storage checks the satisfiability of all the four equations. If the check passes, CS is convinced that the value in the commitment is equal to that in the ciphertext.

4.4.3 *Anonymous Voting*

Based on the previous verification results, CS is convinced that: 1) both the attribute and the corresponding value of voter V are verified by TA; 2) the voter has never used the same attribute to vote the same message before; 3) the encrypted attribute value is the same as the value which has been verified before. Then, the voter is able to vote on the corresponding message using his/her encrypted attribute value. Note that the encryption scheme that we use to encrypt the attribute value has the homomorphic property which is the basic building block in our voting scheme. We now review the homomorphic property of selective-tag encryption.

1) Multiplicative homomorphic property:

$$\begin{aligned}\mathcal{E}(\overline{m}_1)\mathcal{E}(\overline{m}_2) \;=\; &(f^{\overline{r}_1+\overline{r}_2}, h^{\overline{s}_1+\overline{s}_2}, g^{\overline{r}_1+\overline{r}_2+\overline{s}_1+\overline{s}_2}\overline{m}_1\overline{m}_2,\\ &(g^{\overline{t}}\overline{k})^{\overline{r}_1+\overline{r}_2}, (g^{\overline{t}}\overline{l})^{\overline{s}_1+\overline{s}_2}) = \mathcal{E}(\overline{m}_1\overline{m}_2);\end{aligned}$$

2) Self-blinding property:

$$\mathcal{D}(\mathcal{E}(\overline{m}, \overline{r}_1, \overline{s}_1, \overline{t})) = \mathcal{D}(\mathcal{E}(\overline{m}, \overline{r}_1 + \overline{r}_2, \overline{s}_1 + \overline{s}_2, \overline{t})).$$

Once the whole check process has passed, CS uses the ciphertext from V to multiply the known entry where $\mathcal{J} = j$ points to. Since the vote receiver has already uploaded all the ciphertexts as $w_j^* = \mathcal{E}(\sigma_{i.j}^{-1}) := (f^{\overline{r}_e}, h^{\overline{s}_e}, g^{\overline{r}_e+\overline{s}_e}\sigma_{i.j}^{-1}, (g^{\overline{t}\overline{k}})^{\overline{r}_e}, (g^{\overline{t}\overline{l}})^{\overline{s}_e})$ by given different random numbers, we can apply the homomorphic operation on the ciphertext in w_j^* if the vote has the same format. We refer $\sigma_{i.j}^{-1} = g^{-1/(x_i+m_{i.j})}$ which can be easily computed by the vote receiver himself/herself from the given $g^{1/(x_i+m_{i.j})}$. The voter V computes the vote by using the tag $\overline{t}$, the public key $\overline{pk}$ published by the vote receiver and his/her opinion as 1 or -1, all of which result in the final vote as $\mathbb{V} = (f^{\overline{r}_V}, h^{\overline{s}_V}, g^{\overline{r}_V+\overline{s}_V}\sigma_{i.j}g^{\pm 1}, (g^{\overline{t}\overline{k}})^{\overline{r}_V}, (g^{\overline{t}\overline{l}})^{\overline{s}_V})$, where $g^{\pm 1}$ denotes the voting opinion. Note that we use the vote receiver's real ID's $\overline{pk}$ as the public key used to encrypt just for the ease of illustration, while we also give the full design for anonymous voting in Subsection 5.1. The above vote can be computed via homomorphic operation as $\mathbb{V} = \mathcal{E}(\sigma_{i.j})\mathcal{E}(g^{\pm 1})$, in which a voter can send them separately to CS. Then, CS performs the homomorphic operation on the entry with ciphertext c_j and $\mathbb{V}$, where the output is $(\overline{f}^{\overline{r}}, \overline{h}^{\overline{s}}, g^{\overline{r}_V+\overline{s}_V}g^{\pm 1}, (g^{\overline{t}\overline{k}})^{\overline{r}}, (g^{\overline{t}\overline{l}})^{\overline{s}})$ if the voting attribute values $m_{i.j}$ are identical. Otherwise, it outputs an arbitrary bit string. After this operation, CS moves and collects all the operated entries in a protected poll for users in the system to query. Note that the protection poll can be retrieved via the keyword $\mathcal{G}_{K_2}(x)$ and it prohibits any voter from voting using the entries which have been operated before. To avoid the possible tracing attack launched by the vote receiver, CS randomly permutes the data file stored in the protection poll once a new vote comes in. Until now, CS collects all users' voting on the particular message and can further respond to any query intending to retrieve the reputation value for this vote receiver on this attribute and the corresponding message.

4.5 Attribute-Based Reputation Value Retrieval

This protocol is mainly executed by a querier and CS. The major design purpose is applying the proxy re-encryption scheme to re-encrypt the ciphertext in CS in order to let queriers decrypt the reputation values. Different from several existing reputation schemes which enable the vote receiver or collector to publish the reputation value, our scheme only allows users to retrieve reputation value via the centralized storage, where the vote receiver cannot forge or hide the reputation value to the public. To avoid extra computation and storage cost, we let the querier use the real ID to retrieve the reputation value instead of using assigned pseudonyms. Since the reputation is publicly known to every user in the system, we allow a querier to use the real ID to obtain reputation values. In what follows, we enable the user R_1 to request R_2 for the reputation value on the attribute x_r.

4.5.1 Re-Encryption Table Establishment

Our reputation value retrieval scheme relies on the proxy re-encryption for selective-tag encryption scheme. For each entry in CS, we render a re-encryption table to respond to the reputation request launched by valid users in the system. Note that each entry represents one particular user's attribute (vote receiver), and we may have different entries which belong to one user. When a querier launches the reputation retrieval protocol, we can directly re-encrypt a ciphertext in the protection poll without considering a vote receiver or a querier's identity. The re-encryption table is given by TA before the protocol run, and it consists of all of users in the system with real identity. Furthermore, we can easily extend it to anonymous retrieval by redefining the re-encryption table as a matrix. Here, we only consider the real identity retrieval for the ease of description. Take users R_1 and R_2 as an example, and the re-encryption table for R_2 is stored on every entry in CS corresponding to different attributes. The proxy re-encryption key for R_2 is generated as follows, as done in Subsection 4.4.2.

Re-Encryption Key Generation (ReKeyGen): on the input of $\overline{sk}_{R_1} = (\chi_{R_1}, \psi_{R_1})$ and $\overline{sk}_{R_2} = (\chi_{R_1}, \psi_{R_2})$, TA outputs re-encryption key for R_2 as $rk_{R_2\to R_1} = (\chi_{R_1}/\chi_{R_2} \mod p, \psi_{R_1}/\psi_{R_2} \mod p)$.

Re-Encryption(ReEnc): on input the ciphertext $(\overline{f}^{\overline{r}}_{R_2}, \overline{h}^{\overline{s}}_{R_2}, g^{\overline{r}+\overline{s}}g^{\pm 1}, (g^{\overline{t}\overline{k}})^{\overline{r}}, (g^{\overline{t}\overline{l}})^{\overline{s}})$ encrypted under $\overline{pk}_{R_2}$, CS re-encrypts the ciphertext using $rk_{R_2\to R_1}$, where the public key turns out to be $\overline{pk}_{R_1} := ((\overline{f}^{\overline{r}}_{R_2})^{rk_{R_2\to R_1}}, (\overline{h}^{\overline{s}}_{R_2})^{rk_{R_2\to R_1}}) = (g^{(\chi_{R_2}\overline{r})(\chi_{R_1}/\chi_{R_2})}, g^{(\psi_{R_2}\overline{s})(\psi_{R_1}/\psi_{R_2})}) = (g^{\chi_{R_1}}, g^{\psi_{R_1}})$. Then, it outputs the ciphertext as $(\overline{f}^{\overline{r}}_{R_1}, \overline{h}^{\overline{s}}_{R_1}, g^{\overline{r}+\overline{s}}g^{\pm 1}, (g^{\overline{t}\overline{k}})^{\overline{r}}, (g^{\overline{t}\overline{l}})^{\overline{s}})$.

For the user R_2, given all valid users in the system, TA renders the re-encryption keys rk corresponding to users' identities and stores them as a table on CS with different attributes.

4.5.2 Reputation Retrieval Request

User R_2 has published several messages associated with the attribute x_r. To guarantee the identity privacy, R_2 uses different pseudonyms to publish those messages, but he/she also wants to collect the reputation on his/her attribute x_r. After voters vote on those messages, user R_1 wants to know a user PS_{R_2}'s reputation value on the attribute x_r when he/she posts a message and associates it with x_r. The process is as follows,

1) $R_1 \rightarrow PS_{R_2} : RepReq, x_r, \tilde{t}_2, SIG(x_r||\tilde{t}_2)$,

2) $PS_{R_2} \rightarrow R_1$:

$$E_{CS}(\mathcal{G}_{K_2}(x_r)), \tilde{t}_3, SIG(E_{CS}(\mathcal{G}_{K_2}(x_r))||\tilde{t}_3),$$

3) $R_1 \rightarrow CS : \mathsf{Retrieval}(T, E_{CS}(\mathcal{G}_{K_2}(x_r)), R_1), \tilde{t}_4$.

Then, CS runs the protocol ReEnc to re-encrypt all the ciphertext in the protection poll. Finally, CS returns the re-encrypted data file to R_1.

4.5.3 Reputation Value Derivation

The reputation value derivation is run by querier R_1 when he/she obtains a set of ciphertext encrypted by $\overline{pk}_{R_1}$. Given a set of ciphertexts $\{\mathcal{E}^i_{\overline{pk}_{R_1}}(g^j)|j \in \{-1, 1\}\}$, querier R_1 is able to invoke the protocol Dec to decrypt the ciphertext as $g^{\overline{r}+\overline{s}} g^j (\overline{f}^{\overline{r}}_{R_1})^{-1/\chi_{R_1}} (\overline{h}^{\overline{s}}_{R_1})^{-1/\psi_{R_1}} = g^j$ by using $\overline{sk}_{R_1} = (\chi_{R_1}, \psi_{R_1})$. Note that i denotes the index of the ciphertexts. Thus, R_1 can compute and collect the numbers of g and g^{-1} to obtain the reputation value of R_2 on the attribute x_r. We also further analyze the reputation value in detail. For example, if the obtained reputation value is $\Omega := |g_i| - |g_i^{-1}|$, we cannot induce the reputation value for R_2 on a specific attribute in Ω. However, based on the number of messages that R_2 publishes during the protocol run, we need to reconsider the "self-voting" problem in the reputation value derivation process. If a vote receiver publishes multiple messages associated with one specific attribute, we cannot accept all the voting results based on Ω. Since the voter needs to present $U = \hat{H}(vk_{R_2}||\mathcal{M}||L(x)||seq)$ before the voting, we can further enforce CS to record the number of U and render the number of U to a querier during the reputation value retrieval. Thus, the trust range of reputation value could be clarified as $[\Omega - |U|, \Omega]$.

5 Performance Analysis and Discussion

This section studies how the security objectives are achieved in our proposed protocols based on the adversary model, followed by the efficiency analysis on the computational load of our proposed scheme. We also elaborate the discussion on the feasibility of the proposed scheme in real online social networks.

5.1 Security Analysis

5.1.1 Voter Anonymity

Voter anonymity requires that the voter's identity and attribute privacy should be well preserved during the attribute verification, voting privilege check, anonymous voting and reputation retrieval processes. In what follows, we give the detailed analysis among these four steps together with the possible attacks launched by adversaries.

Attribute Verification. Voter's identity privacy is well preserved since we offer every valid user a set of collision-resistant pseudonyms to use during the voting process. Voters may change the pseudonyms when they request the voting privileges even to the same vote receiver. For the attribute privacy, performing as a verifier, a vote receiver learns nothing during the verification process because the verifier is not able to open the commitments in the packet P. On the other hand, the most possible attack is the impersonation attack in which the adversary uses others' credentials. To trace back the identity of the adversary, the only possible way is to call for TA to perform as an arbiter to open the commitments. Suppose we have a suspected user with a commitment $Com(b)$, TA can use the system parameter (α, β) to open the commitment as follows: $(f^{r_1+r_3r_u})^{-1/\alpha}(h^{r_2+r_3s_v})^{-1/\beta} b g^{r_1+r_2+r_3(r_u+s_v)} = b$. Then, together with the pseudonym that this suspected user used, we can locate his/her real identity and find out the user who leaks the corresponding credential, since the value of $\hat{r} \in Z_p$ represents the uniqueness of each user's credential. Also, according to the property of the *soundness* of NIWI proof, malicious users who use others' commitments and the corresponding proof π for verification fail to generate a distinct packet P with the same value inside the commitments, which implies the probability that the malicious user can perform as a valid prover is negligible.

Voting Privilege Check. Since this process only involves voters and CS, we consider the "double-voting" attacks launched by malicious voters and the information collected by CS. Clearly, if a voter follows the designed protocol for the privilege check, the identity would not be discovered by the "double-voting" check, where the voter anonymity is achieved. However, a

malicious user who wants to vote the same attribute-associated message twice can be easily identified and further revoked. For CS, although it stores each voter's serial number S and voting equation T as a pair, it still fails to get the information of message $\mathcal{M}$, verified certificate $\sigma_{\imath.\jmath}$ and voter's secrets ε and ϖ due to the DLP assumption. During the verification of zero-knowledge proof, it is guaranteed in [33] that the verifier, CS, learns nothing according to $\mathbb{B}$, $\mathbb{C}$, $\mathbb{D}$, $\mathbb{E}$ and $\mathbb{F}$. Thus, CS cannot trace any voter according to the information in Σ', such that the voter anonymity is achieved.

Anonymous Voting. The attribute-based anonymous voting includes the verification of NIWI proofs and the NIZK proofs used to check the equality between the values in the commitments and ciphertexts. Apart from the possible attacks in the attribute verification process, the verification process also provides the certificate privacy on $\sigma_{\imath.\jmath}$ due to the assumptions that DLP is considered as a hard problem and that it is infeasible to open the commitments without being given the system parameters (α, β). On the other hand, although CS would not launch attacks, it still can learn the information during the protocol run. To better preserve the voter anonymity, the vote consists of random numbers $(\overline{r}_V, \overline{s}_V)$ together with the decision $g^{\pm 1}$, such that CS cannot learn neither the real identity of a voter nor the vote decision on a designated message.

Reputation Retrieval. One of the most possible attacks to voter anonymity during the reputation retrieval step would be launched by the vote receiver. This type of attack happens when a vote receiver publishes one new message, and immediately retrieves the reputation after he/she gives the vote entry to an anonymous voter. It is possible for this malicious vote receiver to obtain the voting decision of that voter. However, this kind of attack only tampers the anonymity of vote decision rather than the voter's identity. To some extent, since the voter uses one-time pseudonym and the commitments with random numbers, it is hard for the vote receiver to trace the real identity of the voter. On the other hand, we also need to consider the situation where voters postpone their votes to avoid being identified. Thus, this type of attack would not largely compromise the privacy requirements of our scheme. It is also possible for a voter to vote a message based on his/her verified certificate $\sigma_{\imath.\jmath'} \neq \sigma_{\imath.\jmath}$ (indicating $m_{\imath.\jmath'} \neq m_{\imath.\jmath}$). We distinguish this kind of misbehavior into two distinct ways: malicious voting and unconscious voting. The malicious voter uses incorrect certificate or even dummy bit string to compromise the voting entries. However, the step involving the anonymous voting is followed by the previous verification process. If he/she uses incorrect certificate to vote, it would be easy to be identified by using the previous method. On the other hand, if a voter unconsciously votes the ciphertext in CS, the result for each querier would be $g^{\pm 1} g^{\frac{\tilde{x}_\imath + m_{\imath.\jmath}}{\tilde{x}_\imath + m_{\imath.\jmath'}}}$. According to the DLP assumption, it is also hard for queriers to obtain $\sigma_{\imath.\jmath}$ or $\sigma_{\imath.\jmath'}$ from decryption results even when someone holds the above certificates.

5.1.2 *Vote Receiver Anonymity*

To preserve the receiver anonymity, our scheme has the countermeasures to prevent the vote receiver's identity from being traced during the protocol run. Voters may potentially collect the information of one particular vote receiver and find out the real identity. In our scheme, a vote receiver is free to post attribute-associated messages using different pseudonyms since voters judge the content of the message rather than the content generator. The tracing attack happens during the attribute verification and reputation retrieval steps. In the attribute verification process, the voter can collect the information of encrypted voting entries $E_{pk_{CS}}(\mathcal{F}_{K_1}(x), \mathcal{G}_{K_2}(x))$ and $U = \hat{H}(vk_R||\mathcal{M}||L(x)||seq)$. We consider a malicious voter uses different pseudonyms to query the voting entires based on the same attribute-associated message $\mathcal{M}$. However, since the vote receiver also uses pseudonyms to post messages and gives different voting entries $\mathbb{P}(i)_{i \in L(x)} \oplus \mathcal{F}_{K_1}(x)$, it is still infeasible for the voter or even a group of colluded voters to obtain the detail of vk_R (the only information used to uniquely identify a user in the system) from a set of hash values U with different sequence numbers seq. When malicious queriers retrieve the reputation value of a specific vote receiver, they can trace the identity if retrieval entries remain the same. However, the encrypted forms of the retrieval entries given to each querier are distinct due to the random numbers in the ciphertexts. Another security breach would be the published public keys used to generate the vote. In Section 4, we introduce the anonymous voting scheme by rendering each voter the public key $\overline{pk}$ to every voter in the system. More specifically, we use different public keys based on the vote receiver's pseudonyms to encrypt σ^{-1} in CS. Then, CS can apply the proxy re-encryption scheme to re-encrypt the ciphertext before shifting the results to the protection poll. Note that we also render CS another table with the real ID and users' different $\overline{pk}$ corresponding to the assigned pseudonyms, where it can

re-encrypt the ciphertext under the real ID's public key $\overline{pk}$. Another advanced scheme would be rendering different $\mathcal{F}_{K_1}(x)$ based on different keys K_1 to CS and/or applying different indices in $\mathbb{P}(i)_{i\in L(x)}$, where the vote receiver creates voting entries corresponding to different public keys of the pseudonyms. Then, the voter can follow the same procedure introduced in previous sections to vote on the attribute-associated messages. Since all these entries are indexed by the keyword $\mathcal{G}_{K_2}(x)$, we can still retrieve the reputation using the previous method. Thus, we can fully achieve the anonymous voting while maintaining the receiver anonymity.

5.1.3 Confidentiality of the Central Storage

For a semi-trusted centralized storage, we have the countermeasures to prevent it from learning anything related to the voting and collecting processes. First of all, we do not store any plaintext to CS, which hides users' identity and voting decisions. Using as our basic building block, the structured encryption provides us the opportunity to hide the information stored in CS while the indices and the confidentiality of the labeled data are well preserved. We enable the permutation before the index storing to the CS, which prevents the CS from learning the linkage between the ciphertext and the index. However, it is the meaningless label that provides us the way to design the fine-grained reputation system, where each meaningless index represents a unique attribute defined by the user him/herself. Our NIWI, NIZK and other zero-knowledge proofs used in the verification process prevent CS from learning credentials and certificates that users have. One of the possible leakage would happen in the anonymous voting and reputation retrieval processes. Since we deploy two tables for the proxy re-encryption, learning the pseudonyms of a voter and a querier would compromise the identity privacy. Thus, it is reasonable to enforce each user to use the pseudonym only once, which may decrease the possibility of privacy leakage. For the proxy re-encryption, except knowing the pseudo-identity of a querier, the content of the ciphertext still remains unknown to CS. Another important design for CS is the randomization of the protection poll. Once CS applies the homomorphic operation on the voting entry and the votes, it moves the results to the protection poll and randomizes it before the reputation retrieval process. By applying this randomization process, we could guarantee that a querier would not be able to learn the detail of the voting decision when there is a large number of votes.

5.2 Efficiency Analysis

5.2.1 Computational Cost

We used the Pairing-Based Cryptography (PBC-0.5.12) library to implement our simulation. We take Tate pairing as our basic pairing operation. The elliptic curve we use for the our scheme is type A. A curve of such type has the form of $y^2 = x^3 + x$. The order of the curve is around 160 bits, as is F_p, the base field. For the experiments on NIWI and NIZK proofs and verification, we use MacBook Pro with an Intel Core 2 Duo 2.8 GHz and 8 GB RAM. We also refer to the work of Meiklejohn *et al.*[41] for the efficiency analysis of e-cash scheme. Since the CS has large enough space for storing votes and the corresponding data (the maximum proof size for the voting privilege check is approximately 22 526 byte $\approx$ 22 MB), we do not consider the storage cost for the CS. For the user side, it is feasible for a user to clear up the proofs after the voting as long as keeping the log files to show the voted messages. Therefore, we only consider the computation cost for a user and CS. During the attribute verification process, the commitments of two variables consist of six group elements and the verification process costs three group operations. The vote receiver also needs to compute the encrypted voting entries using ID-based encryption which incurs at most one pairing computation according to [30, 42]. During the privilege check process, it costs the voter nice multi-base exponentiation operations to generate the commitments and 11 more for the proofs. The CS also costs 11 multi-base exponentiation operations for verifying the proof. In the attribute value check process, the user needs nine group elements to commit three variables, while the server side incurs 18 pairing operations for the verification. For the equality check, the user computes additional nine group elements for commitments in NIZK and five on the ciphertext, where it totally consists of 29 group elements. The central storage takes 27 group operations to verify the above proofs. All the timing reported below is averaged over 100 randomized runs. Note that the prover and the verifier in Fig.2 represent the voter and the CS in the privilege check and anonymous voting process, while they represent the voter (or querier) and the vote receiver in the attribute verification and the reputation retrieval process, respectively.

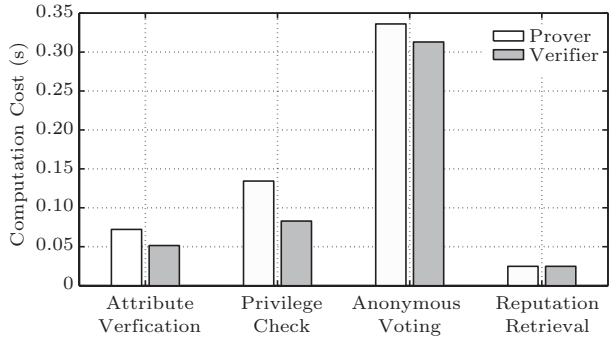


Fig.2. Computation cost analysis.

5.2.2 Comparison with Previous Work

We mainly focus on the comparison with the most similar work done by Benthencourt *et al.* in [11]. Since we only have the anonymous voting and reputation retrieval parts which are the same as the schemes in [11], we focus on the efficiency analysis in these two protocols. In what follows, we consider that the basic operation of proving and verifying one NIZK or NIWI takes up to n group operations (linearly dependent on computation time). As we can see from Table 3, the computational complexity of the two schemes in the voting protocol remains the same. Note that we use λ to represent the number of votes. Our scheme applies NIZK and NIWI proofs to prove the validity of the vote V, which requires $O(n)$ time to compute and verify the corresponding proofs. Although the voting scheme in [11] consists of more sentences *st* to get verified, it also requires $O(n)$ time to generate the NIZK proofs for a single vote. Thus, the total cost of the computation during the voting process is $O(\lambda \times n)$. For the reputation retrieval, our scheme outperforms the signing and verification process in [11]. Since they incorporate the nested NIZK proofs in the verification of signature of reputation, the verification process requires $O(\lambda \times n^2)$ time. However, our scheme only requires $O(\lambda \times n)$ for generating and verifying the corresponding attributes. We implement a hash table to search entries in $\mathcal{T}$, which incurs average $O(\lambda \times n) + O(1 + |I|/N)$ for searching a particular entry. Note $|I|$ denotes the total number of users' verified attributes stored in CS. Since we use kw as an index for searching voting entries, we may use those kw as keys in the hash table to achieve the optimal computation complexity $O(1 + |I|/N)$. Moreover, we can also apply the binary tree to represent the data type of the dictionary $\mathcal{T}$, which requires average $O(\log |I|)$ time for searching a particular entry. Both of the approaches outperform the schemes in [11] in terms of computational complexity.

Table 3. Efficiency Performance Comparison

	Voting	Reputation Retrieval
Benthencourt *et al.*[11]	$O(\lambda \times n)$	$O(\lambda \times n^2)$
Our scheme	$O(\lambda \times n)$	$O(\lambda \times n) + O(1 + \frac{\lvert I \rvert}{N})$

5.3 Discussion on Deployment for OSNs

Our scheme could be directly applied to many OSNs for improving the credibility and privacy of the reputation management system. As one of the examples, LinkedIn has implemented the reputation system named as "endorsement". However, LinkedIn cannot achieve the anonymous voting, which sometimes prevents users from giving negative score on their messages or abilities. Also, most of the endorsements can only achieve the monotonic reputation values, and users cannot give negative values, which may not be ideal for some cases. Our schemes leverage attributes to achieve fine-grained reputation values, where users could register different types of attributes with OSN providers, and use our anonymous voting to endorse the votes (both positive and negative). Finally, for users who want to check someone's reputation values, they can simply launch requests to the centralized storage and obtain the designated users' objective reputation values.

6 Conclusions

In this paper, we proposed a privacy-preserving attribute-based reputation system for online social networks. Our system verifies user-centric attributes for which users want to establish and maintain the reputation. Contrary to traditional reputation systems, our approach offers users a possible way to collect non-monotonic reputation without revealing the identity. To guarantee the authenticity of the reputation value which only relies on the content rather than the content generator, our scheme leverages zero-knowledge proofs to let authorized users use verified attribute values to vote, not users with unverified/unauthorized attributes. Based on the security and efficiency analysis, we showed both the privacy-preservation and the practicality of our proposed scheme.

References

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

[2] Cho J H, Swami A, Chen I R. A survey on trust management for mobile ad hoc networks. *IEEE Communications Surveys & Tutorials*, 2011, 13(4): 562-583.

[3] Sun Y, Yu W, Han Z *et al*. Information theoretic framework of trust modeling and evaluation for ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 2006, 24(2): 305-317.

[4] Theodorakopoulos G, Baras J. On trust models and trust evaluation metrics for ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 2006, 24(2): 318-328.

[5] Guo L, Zhu X, Zhang C *et al*. A multi-hop privacy-preserving reputation scheme in online social networks. In *Proc. IEEE GLOBECOM*, Dec. 2011.

[6] Jøsang A. An algebra for assessing trust in certification chains. In *Proc. NDSS*, Feb. 1999.

[7] Capkun S, Buttyan L, Hubaux J P. Self-organized public-key management for mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 2003, 2(1): 52-64.

[8] Zhang C, Song Y, Fang Y. Modeling secure connectivity of self-organized wireless ad hoc networks. In *Proc. the 27th IEEE INFOCOM*, April 2008.

[9] Guo L, Zhang C, Fang Y. A trust-based privacy-preserving friend recommendation scheme for online social networks. *IEEE Transactions on Dependable and Secure Computing* 2014, PP(99): 1.

[10] Lin P, Chung P C, Fang Y. P2P-iSN: A peer-to-peer architecture for heterogeneous social networks. *IEEE Networks*, 28(1): 56-64.

[11] Bethencourt J, Shi E, Song D. Signatures of reputation. In *Proc. the 14th Int. Conf. Financial Cryptography and Data Security*, Jan 2010, pp.400-407.

[12] Lin P, Chen H, Fang Y *et al*. A secure mobile electronic payment architecture platform for wireless mobile networks. *IEEE Transactions on Wireless Communications*, 2008, 7(7): 2705-2713.

[13] Androulaki E, Choi S G, Bellovin S M, Malkin T. Reputation systems for anonymous networks. In *Proc. the 8th International Symposium on Privacy Enhancing Technologies*, July 2008, pp.202-218.

[14] Groth J. Evaluating security of voting schemes in the universal composability framework. In *Proc. the 2nd ACNS*, June 2004, pp.46-60.

[15] Groth J. Non-interactive zero-knowledge arguments for voting. In *Proc. the 3rd ACNS*, June 2005, pp.467-482.

[16] Camenisch J, Kohlweiss M, Soriente C. An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In *Proc. the 12th Int. Conf. Practice and Theory in Public Key Cryptography*, March 2009, pp.481-500.

[17] Belenkiy M, Camenisch J, Chase M *et al*. Randomizable proofs and delegatable anonymous credentials. In *Proc. the 29th Annual International Cryptology Conference on Advances in Cryptology*, Aug. 2009, pp.108-125.

[18] Blum M, Feldman P, Micali S. Non-interactive zero-knowledge and its applications. In *Proc. the 20th Annual ACM Symposium on Theory of Computing*, Jan. 1988, pp. 103-112.

[19] Groth J, Sahai A. Efficient non-interactive proof systems for bilinear groups. In *Proc. the 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, April 2008, pp.415-432.

[20] Bethencourt J, Shi E, Song D. Signatures of reputation: Towards trust without identity. In *Proc. the 14th International Conference on Financial Cryptography and Data Security*, Jan. 2010.

[21] Groth J. Fully anonymous group signatures without random oracles. In *Proc. the 13th ASIACRYPT*, Dec. 2007, pp.164-180.

[22] Belenkiy M, Chase M, Kohlweiss M *et al*. Noninteractive anonymous credentials. *IACR Cryptology ePrint Archive*, 2007. http://eprint.iacr.org/, Jan. 2015.

[23] Guo L, Zhang C, Sun J *et al*. PAAS: Privacy-preserving attribute-based authentication system for eHealth networks. In *Proc. the 32nd IEEE ICDCS*, June 2012, pp.224-233.

[24] Guo L, Zhang C, Sun J *et al*. A privacy-preserving attribute-based authentication system for mobile health networks. *IEEE Transactions on Mobile Computing*, 2014, 13(9): 1927-1941.

[25] Guo L, Zhang C, Yue H *et al*. A privacy-preserving social-assisted mobile content dissemination scheme in DTNs. In *Proc. IEEE INFOCOM*, April 2013, pp.2301-2309.

[26] Bethencourt J, Sahai A, Waters B. Ciphertext-policy attribute-based encryption. In *Proc. IEEE Symposium on Security and Privacy*, May 2007, pp. 321-334.

[27] Barua M, Liang X, Lu R *et al*. PEACE: An efficient and secure patient-centric access control scheme for eHealth care system. In *Proc. IEEE INFOCOM WKSHPS*, April 2011, pp.970-975.

[28] Narayan S, Gagné M, Safavi-Naini R. Privacy preserving EHR system using attribute-based infrastructure. In *Proc. ACM Workshop on Cloud Computing Security Workshop*, Oct. 2010, pp.47-52.

[29] Barić N, Pfitzmann B. Collision-free accumulators and fail-stop signature schemes without trees. In *Proc. the International Conference on the Theory and Application of Cryptographic Techniques*, May 1997, pp.480-494.

[30] Boneh D, Franklin M. Identity-based encryption from the Weil pairing. In *Proc. the 21st CRYPTO*, Aug. 2001, pp.213-229.

[31] Schnorr C P. Efficient signature generation by smart cards. *J. Cryptology*, 1991, 4(3): 161-174.

[32] Camenisch J, Michels M. Proving in zero-knowledge that a number is the product of two safe primes. In *Proc. the International Conference on the Theory and Application of Cryptographic Techniques*, May 1999, pp.107-122.

[33] Camenisch J, Stadler M. Efficient group signature schemes for large groups (extended abstract). In *Proc. the 17th Annual International Cryptology Conference on Advances in Cryptology*, Aug. 1997, pp.410-424.

[34] Chase M, Kamara S. Structured encryption and controlled disclosure. In *Proc. the 16th International Conference on the Theory and Application of Cryptology and Information Security*, Dec. 2010, pp.577-594.

[35] MacKenzie P, Reiter M K, Yang K. Alternatives to nonmalleability: Definitions, constructions, and applications (extended abstract). In *Proc. the 1st TCC*, Feb. 2004, pp.171-190.

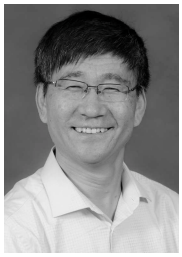
[36] Boneh D, Boyen X, Shacham H. Short group signatures. In *Proc. the 24th CRYPTO*, Aug. 2004, pp.41-55.

[37] Boneh D, Boyen X. Short signatures without random oracles and the SDH assumption in bilinear groups. *Journal of Cryptology*, 2008, 21(2): 149-177.

[38] Camenisch J, Hohenberger S, Lysyanskaya A. Compact e-cash. In *Proc. the 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, May 2005, pp.302-321.
[39] Camenisch J, Lysyanskaya A. A signature scheme with efficient protocols. In *Proc. the 3rd Int. Conf. Security in Communication Networks*, Sept. 2002, pp.268-289.
[40] Fiat A, Shamir A. How to prove yourself: Practical solutions to identification and signature problems. In *Proc. CRYPTO*, May 1986, pp.186-194.
[41] Meiklejohn S, Erway C, Kupcu A *et al.* ZKPDL: A language-based system for efficient zero-knowledge proofs and electronic cash. In *Proc. the 19th USENIX Conference on Security*, Aug. 2010.
[42] Hess F. Efficient identity based signature schemes based on pairings. In *Proc. the 9th Annual International Workshop on Selected Areas in Cryptography*, Aug. 2002, pp.310-324.

Linke Guo received his B.E. degree in electronic information science and technology from Beijing University of Posts and Telecommunications, Beijing, in 2008. He received his M.S. and Ph.D. degrees in electrical and computer engineering from the University of Florida in 2011 and 2014, respectively. He has been an assistant professor in the Department of Electrical and Computer Engineering, Binghamton University, since August 2014. His research interests include network security and privacy, social networks, and applied cryptography. He has served as the technical program committee member for several conferences including IEEE INFOCOM, ICC, GLOBECOM, and WCNC. He is a member of IEEE and ACM.

Chi Zhang received his B.E. and M.E. degrees in electrical and information engineering from Huazhong University of Science and Technology, Wuhan, in 1999 and 2002, respectively, and Ph.D. degree in electrical and computer engineering from the University of Florida in 2011. He joined the University of Science and Technology of China in September 2011 as an associate professor of the School of Information Science and Technology. His research interests are in the areas of network protocol design and performance analysis, and network security particularly for wireless networks and social networks. He has published over 60 papers in journals such as IEEE/ACM Transactions on Networking, IEEE Journal on Selected Areas in Communications, and IEEE Transactions on Mobile Computing, and in networking conferences such as IEEE INFOCOM, ICNP, and ICDCS. He has served as the technical program committee member for several conferences including IEEE INFOCOM, ICC, GLOBECOM, WCNC and PIMRC. He is the recipient of the 7th IEEE ComSoc Asia-Pacific Outstanding Young Researcher Award.



Yuguang Fang received his Ph.D. degree in systems engineering from Case Western Reserve University in January 1994 and Ph.D. degree in electrical engineering from Boston University in May 1997. He was an assistant professor in the Department of Electrical and Computer Engineering at New Jersey Institute of Technology from July 1998 to May 2000. He then joined the Department of Electrical and Computer Engineering at University of Florida in May 2000 as an assistant professor, and got an early promotion to an associate professor with tenure in August 2003 and a professor in August 2005. He has published over 350 papers in refereed professional journals and conferences. He received the National Science Foundation Faculty Early Career Award in 2001 and the Office of Naval Research Young Investigator Award in 2002. He won the Best Paper Award at IEEE ICNP 2006. He has served on many editorial boards of technical journals including IEEE Transactions on Communications, IEEE Transactions on Wireless Communications, IEEE Transactions on Mobile Computing, Wireless Networks, and IEEE Wireless Communications (including the Editor-in-Chief). He is also serving as the technical program co-chair for IEEE INFOCOM 2014. He is a fellow of IEEE.

Phone Lin is a professor in "National" Taiwan University, Taipei, holding professorship in the Department of Computer Science and Information Engineering, Graduate Institute of Networking and Multimedia, Telecommunications Research Center, and Optoelectronic Biomedicine Center. Lin serves on the editorial boards of several journals such as IEEE Transactions on Vehicular Technology, IEEE Wireless Communications Magazine, and IEEE Internet of Things Journal. He has also been involved in several prestigious conferences, such as holding co-chair of the Wireless Networking Symposium, IEEE Globecom 2014. Lin has received numerous research awards such as Junior Researcher Award of Academia Sinica in 2010, Ten Outstanding Young Persons Award of Taiwan in 2009, and Best Young Researcher of IEEE ComSoc Asia-Pacific Young Researcher Award in 2007. Lin is an IEEE senior member and an ACM senior member. He received his B.S. and Ph.D. degrees in computer science and information engineering from "National" Chiao Tung University, Hsinchu, in 1996 and 2001, respectively.