ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
DOI: 10.17559/TV-20150314122306

MAKING REPUTATION SYSTEM TRACEABLE WITHOUT LOSING PRIVACY

Keli Zhang, Yi Xian Yang

Preliminary note

In existing reputation system nodes usually adopt regular pseudonyms instead of true identities to gain the anonymity. However complete anonymity will cause watershed and Sybil attack, which look on system be out of control and break the fairness of the reputation system. This paper introduces the conditional anonymity mechanism to check the evaluation between anonymity nodes, evaluation between two nodes if not more than the several times, but not be evaluated between two pseudonyms belong to the same peer. It will be effective, otherwise, the node true identity will be exposed, if number of attacks of the peer within the time exceeds *d* times, nodes in the evaluation and transaction will be tracked. Analysis shows that this mechanism cannot only protect the identity of peer's anonymity, but identify and track malicious attackers.

Keywords: *conditional anonymity; privacy; pseudonym; reputation system; traceable*

Izrada sljedivog sustava ugleda bez gubitka privatnost

Prethodno propćenje

U postojećim čvorovima sustava ugleda obično se prihvaćaju redoviti pseudonimi umjesto pravog identiteta radi postizanja anonimnosti. Međutim, potpuna anonimnost će uzrokovati prekretnicu i Sybil napad, koji bi sustav ugleda stavio izvan kontrole i razbio pravičnost sustava ugleda. Ovaj rad predstavlja uvjetni mehanizam anonimnosti za provjeru vrednovanja između anonimnosti čvorova, vrednovanja između dva čvora ako se ne ponavlja nekoliko puta, ne može se vrednovati između dva pseudonima koji pripadaju istoj razini. On će biti učinkovit, ako će pravi identitet čvora biti izložen, ako je broj napada iste razine unutar vremena koje prelazi *d* vremena, čvorovi u procjeni i transakciji će biti praćeni. Analiza pokazuje da ovaj mehanizam ne samo da može zaštititi identitet istorazinske anonimnosti, već će identificirati i pratiti zlonamjerne napadače.

Ključne riječi: *privatnost; pseudonim; sljediv; sustav ugleda; uvjetna anonimnost*

1 Introduction

The basic idea of trust and reputation model is to allow mutual evaluation between both sides of a deal [1]. The evaluation will help other nodes determine whether to make a deal with the node being evaluated. The reputation of a user depends entirely on its identity. In order to accumulate its reputation value, a user will normally use its real identity or use a pseudonym for a long time. But a long-term use of the same identity will link all the exchange information of a user together. Malicious users or outlaws will systematically analyse the data. All the activities and hobbies of a user will be exposed. This will pose a grave threat to the privacy of users [2].

In order to ensure complete anonymity, users need to change their pseudonyms periodically, which will incur the conflict between privacy and reputation, render reputation evaluation uncontrollable, and reduce the accountability of the reputation system. Truthful users may suffer from malicious users while malicious users can go unpunished. It is because malicious users abandon their previous pseudonyms along with the reputation values, and regain a new one with an initial value. Besides, due to lack of controllability, two users can collude with each other and make false mutual evaluations to increase their reputation value; malicious users will generate many pseudonyms to evaluate themselves, increasing their reputation value.

To solve the above-mentioned problems, this paper introduces evaluative mechanism, conditional anonymity in reputation system. Each user will first acquire an evaluation container from TTP based on CL protocol, and generate a Rater certificate and a Ratee certificate. When the user submits its reputation value to TTP, Rater certificate and Ratee certificate will be submitted as well. Evaluation certificates will anonymously monitor reputation evaluations between nodes. Only if the following anonymity conditions are met: 1) no ramming attack or aspersion; 2) no Sybil attack; can a node gain the reputation value, otherwise, the real identity of a node will be exposed. If a node is attacked d times or more in a limited amount of time, all evaluation and exchange information of the node will be traced. In this protocol, reputation value is bound to the fixed identity (public key or other permanent identity) to strike a balance between anonymity and reputation and ensure the privacy of identity when the reputation value is updated based on blind signature scheme.

2 Related work

At present, among researches on reputation system, most focus on building distributed reputation systems rather than worrying about privacy [3, 4]. In [5÷7], the works only address the conflict issue of anonymity in reputation system. In [8, 9], users use vague identities, but all their activities will still be linked. Authors [2, 10] allow a user to have multiple pseudonyms, but transaction information of each pseudonym will be linked by attackers. In [11, 12], peers gain anonymity through concealing their pseudonyms. But these systems still cannot ensure completely anonymity of users. In [13, 17], users periodically change their pseudonyms to ensure the anonymity of their identity and cut the links between pseudonyms. But in [13], the author only ensures that the e-cash as the reputation value cannot be repeatedly used, but does not control the frequency of evaluation for the two pseudonyms. In [17], the author introduces two TTPs to control the frequency of evaluations between two users, but the real identity of the two users is exposed to a third-party, secure multiparty computation [14] and Homomorphic encryption [16] is proposed to protect the identity of rater, but the protocol cannot to identity and

resist the Malicious attacks. Recently a new cryptographic primitive called signature of reputation was proposed in [15].

Conditional anonymity remains an important research subject. At present, research is mainly focused on e-cash [18, 19]. Another kind of research is focused on key escrow [20, 21]. Recent correlation study [22, 23] is about a user being able to only use *k* anonymous and unlinked identities in a limited amount of time. Conditional anonymity in reputation system is rarely touched upon in research.

3 Preliminaries

Preliminaries are classified into bilinear map, complexity assumptions and cryptographic tool. The main classification is as follows:

3.1 Bilinear map

Assume G_1 , G_2 , G_T are cyclic groups with the exponent being prime number q, and g_1 is the generator of G_1, g_2 the generator of G_2; η is the isomorphism that can realize the calculation from G_2 to G_1, $\eta(g_2)=g_1$; e is a bilinear map: $e: G_1 \times G_2 \to G_T$. If the following condition is met, then the group (G_1, G_2) is a bilinear group: 1) bilinearities: $g_1 \in G_1$, $g_2 \in G_2$, $a,b \in Z_q$, $e(g_1^{\ a}, g_2^{\ b}) = e(g_1, g_2)^{ab}$ are given; 2) non-degeneracy: $e(g_1, g_2) \neq 1$; 3) computability: map e can effectively compute any input pairs.

3.2 Complexity assumptions

- **Strong RSA assumption**

Strong RSA assumption [24]: Assuming $n = pq$ is the RSA modulus, ($p = 2p'+1$, $q = 2q'+1$ and p, p', q' are prime numbers) component $g \in Z_n^*$, when $e > 1$ and the condition $h^e \equiv g \bmod n$ is met, it is difficult to find a two-tuples (h, e).

3.3 Cryptographic tool

- **CL signature**

Camenish-Lysyanskaya (CL signature) [25] scheme is based on strong RSA assumption. It is between a user and a signer, allowing the user to acquire a signature σ about the question C (C is a specific question about $(v_1,...,v_l)$). But the signer does not know the exact value of C. The signer calculates $CLSign(C)$ and sends it to the user, who will get $\sigma = sign(v_1,...,v_l)$, and be able to verify the signature through $Verif(\sigma,(v_1,...,v_l)) = 1$. When the user verifies the signature, besides giving necessary information to the signer, he also has to send to the signer the signature of knowledge that holds information on the value of C $(v_1,...,v_l)$.

- **DY Verifiable Random Function**

Assume $G =< g >$ is a group with the exponent being q (a large prime number), $s \in Z_q$. This model adopts verifiable random function $F_{g,s}^{\ DY}(x) = g^{1/(x+s+1)}$ $x \in Z_q^*$ put forward by Dodia and Yampolskiy [26]. This function is secure under Y-DDHI.

- Pedersen commitment
- Pedersen [27] puts forward a commitment scheme on discrete logarithm problem. The public parameter is the prime exponent q of the group. The generated components $(g_0,\ldots, g_m)$, and commitment components $(v_1,...,v_m) \in Z_q^{\ m}$. Choose a random number $r \in Z_q$,
- $C = PedCom(v_1,...,v_m;r) = g_0^{\ r} \coprod_{i=1}^{m} g_i^{\ v^i}$.

- **Blind signature**

D. Chuam first put forward the term blind signature [28]. Blind signature means that someone needs to sign for some data, unaware of the contents of the data. Compared with normal digital signatures, blind digital signature has two prominent features: (1) the contents of the information are not known to the signer; (2) after the signature is publicized by the recipient, the signer cannot trace the signature.

- **Known discrete-logarithm-based, zero-knowledge proofs**

This paper uses the identification of zero-knowledge proof based on discrete logarithm put forward by Camenisch and Stadler [30]. For example: $PK\{(\partial,\beta,\delta): y = g^{\partial}h^{\beta} \wedge \widetilde{y} = \widetilde{g}^{\partial}\widetilde{h}^{\beta}\}$, denotes the zero-knowledge proof of ∂,β,δ , and $y = g^{\partial}h^{\beta}$, $\widetilde{y} = \widetilde{g}^{\partial}\widetilde{h}^{\beta}$ are tenable, where y, g, h, $\widetilde{y}$, $\widetilde{g}$, $\widetilde{h}$ are components of $G =< g >=< h >$ and $\widetilde{G} =< \widetilde{g} >=< \widetilde{h} >$. In the meantime, Fait-Shamir [29] heuristic can be used to turn zero-knowledge proof into knowledge signature of m , written as: $SPK\{(\alpha): y = g^{\partial}\}(m)$.

4 General descriptions

This chapter will introduce assumptions of the system, participants and general designing ideas, etc.

4.1 Assumptions

Network environment: Assume the whole reputation system is situated in anonymous network (an Onion Router, Mixnet, etc.).

Pseudonym: Many pseudonyms will be deposited at every node in advance. The validity of pseudonyms will be verified by TTP.

4.2 Participating entities

Node(User): A node is a normal node in the network. Each node plays two different roles: as a Rater, it gives

The evaluation values to other nodes, or as a Ratee, it receives others' evaluation of itself. Transaction and evaluation between two nodes is realized through pseudonym, which is also used to show the reputation value to other users.

TTP (Trusted Third Party): TTP controls information related to the reputation of each node. It creates a reputation account for each user based on the user's public key. It is assumed that TTP is trusted when verifying pseudonyms and calculating the reputation value, but not trusted when it comes to user's privacy.

4.3 General designing ideas

Assume the CL signature key pair of TTP and user's key pair are (pk_{TTP}, sk_{TTP}) and (pk_u, sk_u) respectively, where $pk_u = g^{sk_u}$. $F_{g,s}^{DY}$ is the DY pseudo-random function. Each user will first anonymously acquire a certificate container from TTP. The container consists of the key seed s and a user's key sk_u of $F_{g,s}^{DY}$, and the CL signature of TTP (sk_u, s). CL signature ensures that TTP will not know s and sk_u. After a deal is over, Rater first generates evaluation certificate $Cert_{rater}$ and submits it to Ratee. $Cert_{rater}$ consists of: tag of the certificate $S = F_{g,s}^{DY}(R_1)$, double evaluation equation $T = pk_u F_{g,s}^{DY}(R_2)^r$ and ZKPOK ϕ (which proves that S and T are validly generated by $\sigma_{TTP}(sk_u, s)$). R_1 and R_2 are related to Ratee's pseudonym. Subsequently, Ratee generates Ratee certificate $Cert_{ratee}$, Ratee certificate is similar to evaluative certificate, consisting of the tag of certificate $V = F_{g,s_1}^{DY}(R_1)$, double evaluation equation $W = pk_u F_{g,s_1}^{DY}(R_2)^t$ and ZKPOK φ. R_1 and R_2 are the same with R_1 and R_2 in the evaluation certificate. Ratee submits $Cert_{rater}$ and $Cert_{ratee}$ to TTP. TTP verifies $Cert_{rater}$ and $Cert_{ratee}$. If they are valid, TTP temporarily deposits $Cert_{rater}$ and $Cert_{ratee}$. If they are invalid, TTP abandons $Cert_{rater}$ and $Cert_{ratee}$.

If Rater evaluates the same user twice or more in a limited amount of time, two evaluation certificates $Cert_{rater}$ must use the same evaluation tag S, and the two Ratee certificates $Cert_{ratee}$ must use the same tag V as well. Thus, TTP can acquire the public key of Rater and Ratee based on double evaluation equation: $pk_{rater} = (\frac{T_2^{r_1}}{T_1^{r_2}})^{(r_1 - r_2)^{-1}}$ and $pk_{ratee} = (\frac{W_2^{t_1}}{W_1^{t_2}})^{(t_1 - t_2)^{-1}}$; if both pseudonyms of the same user make evaluations, evaluative certificate tag V and Ratee certificate tag S will be the same. TTP can acquire the public key of the violator $pk = (\frac{T^t}{W^r})^{(t-r)^{-1}}$ based on double evaluation equation T and W. TTP records the identity of the malicious violator that gets caught. When the node updates its reputation value to TTP with the same public key, it must submit a sub-key s_i of the evaluation container. s_i is generated through verifiable key sharing mechanism. Each s_i contains a verifiable secret piece. Once the number of s_i acquired from a certain node by TTP exceeds the threshold, the s of the node's evaluation container will be leaked, and all the node's evaluations will be traced.

5 Security model

5.1 The definition of general security

The protocol is mainly composed of the following protocols and algorithms:

- TTPKeygen(1^k, params) : The key-generation algorithm of TTP. Input the system's public parameter 1^k and the identity of TTP, the output will be the public/private key pair of TTP (pk_{TTP}, sk_{TTP}).
- UserKeyGen(1^k, params) : The key-generation algorithm of the node. Input the system's parameter 1^k and the identity of the node ID, the output will be the public/private key pair of the node (pk_u, sk_u).
- Withdraw$(U(pk_u, sk_u), TTP(pk_{TTP}, sk_{TTP}))$ Evaluation certificate withdrawal protocol. The node acquires evaluation certificate container from TTP. The node's output will be evaluation container W, or false information.
- $RaterCert(U(W, pk_M), M(sk_M, pk_{TTP}, P_M))$ Evaluation certificate-generation protocol. Node U (using pseudonym P_U), based on the evaluation certificate container, generates evaluative certificate $Cert_{rater}$ on node M (using pseudonym P_M). M acquires the evaluation tag S of $Cert_{rater}$ and the validity proof of the identification π.
- $RateeCert(M(W, pk_{TTP}), TTP(sk_{TTP}, pk_{TTP}, P_M))$ Evaluated certificate-generation protocol. Node M (using pseudonym P_M) submits Ratee certificate $Cert_{ratee}$ to TTP. TTP acquires the evaluated identification V of M and the validity proof of the identification φ.
- VerifyRep(M$((S, \pi), (V, \varphi), pk_{TTP}), TTP(pk_{TTP}, sk_{TTP}))$ Reputation evaluation verifying protocol. If Rater and Ratee are both trusted, TTP will accept the Ratee certificate and Rater certificate submitted by Ratee M (using pseudonym P_M).
- RepDeposit(M$(\sigma, pk_{TTP}), TTP(pk_{TTP}, sk_{TTP}))$ Reputation value depositing protocol. If the blind signature is valid, TTP will deposit and update the new reputation value of node M.
- Identity(params, S, π_1, π_2) Attacker identification algorithm. The system can identify the same certificate tag S of the attacker, and the two identification proofs π_1 and π_2. The output of the algorithm is a public key pk_u and the proof Π_G.
- VerifyGuilt(params, S, Π_G, pk_u) Guilt verifying algorithm. It allows publicizing the guilt proof Π_G.

5.1.1 Soundness

Rater can evaluate its deal-maker through the evaluation certificate container. But the frequency of evaluation cannot exceed a limit. A knowledge extractor K is given. K and the adversary execute the certificate withdrawal protocol *Withdraw*, extracting m evaluation certificate tags $S_1, S_2 ..... S_m$. For each evaluation certificate identification of the adversary, when $S \neq S_i \ \forall 1 \le i \le m$, the probability that TTP accepts (S, π) as a valid evaluation certificate is negligible.

5.1.2 Identification of the attacker

Assume TTP is truthful, and U_1 and U_2 are both truthful users. When U_1, U_2 and the adversary execute the evaluation certificate-generation protocol, the output of U_1 will be (S, π_1), the output of U_2 (S, π_2). This guarantees high probability that the public key pk_u and proof Π_G generated from Identity(params, S, π_1, π_2) are accepted by VerifyGuilt(params, S, Π_G, pk_u).

5.1.3 Anonymity

TTP cannot acquire any information on Ratee or Rater even if it colludes with malicious users. This protocol introduces simulator S. S has some information that other participants cannot know. For example: in a normal parameter model, S itself generates information on parameters. In a random oracle model, S is the controlled random oracle. Simulator S should be able to simulate evaluation certificates (Rater certificate and Ratee certificate) without accessing the certificate container. The simulated certificate cannot be distinguished from the valid certificate. To be more specific, S can execute certificate-generation protocol (Rater certificate and Ratee certificate) without knowing secret s of the certificate container or accessing the container.

5.1.4 Excludability

Assume the adversary participated in the execution of the certificate-withdrawal protocol of a truthful node with the public key being pk_u, and subsequently in the execution of certificate-generation protocol of the user. When an attacker inputs a public key certificate pk_u and a forged proof Π_G, and claims the user is the attacker, the probability that forged proof Π_G gets accepted by VerifyGuilt(params, S, Π_G, pk_u) is negligible.

5.2 Definition of formal security

The definition of formal security depends on knowledge extractor K and the simulator S. For a given deal, the knowledge extractor K and extract the evaluated certificate tag of the adversary (when unaware of the node's key or the secret of the certificate container, simulator S) can simulate the evaluation certificate-generation protocol and the evaluated certificate-generation protocol. Under normal circumstances, the definition of knowledge extractor K and the simulator S depends on provable security model. To make the definition more general, a special protocol like a knowledge proof or zero-knowledge proof is needed. For example: a knowledge proof in the withdrawal protocol *Withdraw*, while a zero-knowledge proof in the evaluation certificate (Ratee certificate) generation protocol.

$X - Y$ is used to represent the security model, $X \in (IO, BB, NBB)$, (where IO represents the input-output model, BB represents black-box model, and NBB represents non-black-box model), $Y \in (K, RO)$. The algorithm (Alg) with which Adversary A accesses $X - Y$ model is $A\lg^{X-Y(A)}$. In some models of public-parameter, knowledge extractor K and simulator S are allowed to access some auxiliary information (the auxiliary information will not be provided to participants. The information knowledge extractor K and simulator S access respectively are represented by *auxext* and *auxsim*.

In the $X - Y$ model in the language L, $\mathrm{K}_{\mathrm{Prot},L}^{X-Y(A)}(params, auxext, x)$ represents the knowledge extractor of knowledge proof protocol (Prot), while $S_{\mathrm{Prot},L}^{X-Y(A)}(params, auxsim, x)$ represents zero-knowledge simulator of proof protocol (Prot).

5.2.1 Validity

To make it easier to analyse, *Withdraw* protocol is divided into three parts: $Withdraw_b$, $Withdraw_m$ and $Withdraw_e$, respectively denoting the beginning, the middle and the ending of the protocol (letters b, m, e represents beginning, middle and ending). $Withdraw_b$ ends when the node sends message to TTP. $Withdraw_e$ ends when TTP sends message to the user. $Withdraw_e$ is taken as the proof protocol. The user is the prover, TTP verifier. $Withdraw_m$ output from the verifier determines whether TTP will continue $Withdraw_e$, and send the ending message to the peer. m_1 represents the first information the peer will send to TTP. b_1 represents the state information when TTP received m_1. The detailed requirements of validity are as follows:

a) Whatever model is given, there exists a highly efficient interpretable language L_S and a knowledge extractor $\mathrm{K}_{\mathrm{Prot},L}^{X-Y(A)}(params, auxext, b_1, m_1)$. All b_1 are calculated by TTP. All m_1 are extracted tags $\alpha = (S_1, S_2 ..... S_m, aux)$, $(b_1, m_1, \alpha) \in L_S$. In $Withdraw_m$ the probability that TTP accepts m_1 is not negligible.

b) With regard to input $(params, pk_{TTP})$, adversary *A* executes the game as follows: *A* plays withdrawal protocol with TTP and reputation verifying protocol VerifyRep for any times. If the *i*st withdrawal protocol *Withdraw* succeeds, then the output of knowledge

extractor $K_{Prot,L}^{X-Y(A)}(params, auxext, pk_i, b_{1,i}, m_{1,i})$ will be $(b_{1,i}, m_{1,i}, \alpha_i) \in L_S$; the evaluation identifications list acquired by f executing $Withdraw$ is represented by $A_f = (S_{i,j} | \leq i \leq f, 1 \leq j \leq m)$. For some f, in reputation verifying protocol, if the honest TTP accepts evaluation identification $S \notin A_f$, A wins. It needs to be proven that the probability is negligible that adversary A wins the game within probabilistic polynomial time (PPT).

5.2.2 The identification of attacker

The feature of identifying the attacker ensures that within PPT time, the probability that the adversary A wins the following game is negligible. Public parameter model assumes *params* and *auxext* are fixed.

With regard to input $(params, pk_{TTP})$, adversary A executes the game as follows: A executes $Withdraw$ and reputation verifying protocol VerifyRep with TTP. If the withdrawal protocol is executed successfully, the output of the knowledge extractor $K_{Prot,L}^{X-Y(A)}$ will be $(b_{1,i}, m_{1,i}, \alpha_i) \in L_S$; A_i refers to the evaluation tag of the node with the public key being pk_i . In the reputation verifying protocol VerifyRep , if TTP accepts twice the same evaluation identification $S \in A_i$, the attacker identifying algorithm Identity(Params, S, π_1, π_2) generates pk_i and the proof Π_G . The probability is relatively high that VerifyGuilt(params, S, Π_G, pk_u) accepts the public key pk_i and the proof Π_G .

5.2.3 Anonymity

Assume adversary A inquires whatever information with the public key pk_{TTP} :

a) Public key of the inquiry node j: A requests and accepts the public key pk_j of j, and generates valid and truthful key pair (pk_j, sk_j) .

b) *Withdraw* of node j: A and j execute Withdraw$(U(pk_u, sk_u), A(state))$, *state* refers to the state of A. The *i*st *Withdraw* inquiry is represented by W_i . If W_i is false information, W_i is the invalid certificate container.

c) *RaterCert* of j: A executes RaterCert generation protocol based on the container W: $RaterCert(U(W_j), A(state))$.

If A only makes reputation evaluation based on valid certificate container, but never makes evaluations of violations, then A is valid.

An existing simulator $S^{X-Y(A)(.)}(params, auxsim, .)$ is needed. No valid adversary A can distinguish whether it is playing Game R(Real) or Game I(Ideal):

Game R: The answer to the question A inquires is shown above.

Game I: The public key A inquires and *Withdraw* answer are shown above. With regard to reputation evaluation certificate generation protocol RaterCert , A interacts with simulator $S^{X-Y(A)(.)}(params, auxsim, pk_{TTP})$ not $U(W_j)$.

5.2.4 Excludability

Excludability ensures that only nodes that make malicious attacks need to be punished, while honest nodes need not. There are two kinds of excludability: weak excludability and strong excludability. Weak excludability punishes nodes that make attacks, while strong excludability not only punishes nodes that make attacks, but also makes malicious attackers accountable for their behaviour. This paper considers only weak excludability. To define the weak excludability in this chapter, assume adversary A will make the following attacks on user U:

a) **Initialization** Adversary A acquires the public key of TTP through attacks.

b) **Inquiry** Adversary A inquires node U as such:

Withdraw: The adversary fakes its identity as TTP and executes certificate withdrawal protocol *Withdraw* with the user. The user outputs the evaluation certificate container *W*.

RaterCert: The adversary fakes its identity as Ratee, and executes evaluation certificate generation protocol with Rater. Rater's output is the evaluation certificate container W .

Success criterion: When the protocol ends, if VerifyGuilt(params, S, Π_G, pk_u) accepts adversary A's output (S, Π_G) , then A wins the game.

6 Detailed description of the protocols

This chapter introduces the initialization, evaluation certificate withdrawal protocol (*Withdraw*), evaluation Rater certificate-generation protocol (*RaterCert*), Ratee certificate-generation protocol (*RateeCert*), reputation evaluation verifying protocol (*VerifyRep*), reputation value updating protocol (*RepDepost*), attacker identifying algorithm (*Identity*) and tracing malicious nodes.

6.1 Initialization

- **Instructions on the parameter**

$G_1 =< g_1 >$, n is the modulus of special RSA. g_1 is the quadratic residue modulus n, $h_1 \in G_1$. Pedersen commitment and CL signature based on RSA both use this parameter.

$G_2 =< g_2 >$, g_2 is the prime exponent $q = \Theta(2^k)$, $h_2 \in G_2$. Certificate identification, double evaluation equation both use this parameter and $F_{g,s}^{DY}(x) = g_2^{1/(x+s+1)}$.

- **Key generation**

TTP key generation: by executing TTP key generation algorithm TTPKeygen$(1^k, \text{params})$ g CL signature key pair and blind signature key pair of TTP

(pk_{TTP}, sk_{TTP}) and $(pk_{TTP}{}^{\sigma}, sk_{TTP}{}^{\sigma})$ respectively will be generated;

User's key generation: by executing user's key generation algorithm UserKeyGen(1^k, params), and inputting the system parameter 1^k node's identity ID, user public/private key pair (pk_u, sk_u) will be generated.

6.2 Evaluation certificate container withdrawal protocol

Assume the user and TTP authenticate each other. The protocol with which the user acquires evaluation certificate container from TTP is as follows. The protocol is executed based on CHL e-cash withdrawal protocol:

1) The user uses the real identity to negotiate a commitment value C' with TTP with regard to $s \in Z_q$: the user chooses a random number $s' \in Z_n^*$, and calculates Pedersen commitment $C' = PedCom(sk_u, s'; r) = g_0^r g_1^{sk_u} g_2^{s'} (\text{modn})$. The user sends C' and pk_u to TTP, as well as the corresponding secret s to C' and the zero-knowledge proof of the user's private key sk_u.
2) TTP first verifies the zero-knowledge signature ϕ. If it is correct, TTP chooses a random number $r' \in Z_n^*$, calculates $C = C' g_0^{r'} (\text{modn})$, and executes CL signature $\sigma_{TTP} = C^{1/e} \bmod n = (g_0^{s'} g_1^{sk_u} g_2^{r} g_0^{r'})^{1/e}$. TTP sends σ_{TTP} and r' to the user, and deposits the acquired C, pk_u, ϕ, r', σ_{TTP} into the database, used to identify malicious users.
3) The user calculates $s = (s' + r') \bmod n$, and verifies the CL signature of TTP, or verifies whether the following equation is tenable:

$$\sigma_{TTP}^e = (g_0^{\ s} g_1^{sk_u} g_2^{\ r}) \quad (1)$$

If it is, the user can confirm it is the signature of TTP.
4) The user certificate container is $W = (sk_u, s, \sigma_{TTP}(sk_u, s))$, where s is the secret of the evaluation certificate container, $\sigma_{TTP}(sk_u, s)$ is the signature of TTP. According to CL signature protocol, TTP cannot know the values of sk_u, s.

6.3 Evaluation certificates generation protocol

Two users U and M use P_U and P_M to make a deal. After the deal, P_U evaluates P_M. P_U will generate evaluation certificate $Cert_{rater}$ based on the information on P_M's pseudonym and its own certificate container. The process of P_U generating $Cert_{rater}$ is as follows:

1) P_M calculates $R_1 = H(0 \Pi P_M)$, $R_2 = H(1 \Pi P_M)$; $H(.)$ is a one-way hash function with strong collision resistance;
2) P_M sends the random number r to P_U, where $r \in Z_q$;
3) P_U sends the certificate identification and double evaluation Eq. $S = F_{g,s}{}^{DY}(R_1)$; $T = pk_u F_{g,s}{}^{DY}(R_2)^r$
4) P_U sends ZKPOK ϕ of $(R_1, R_2, sk_u, s, \sigma_{TTP}(sk_u, s))$ to P_M:

- $R_1 = H(0 \Pi P_M)$;
- $R_2 = H(1 \Pi P_M)$;
- $S = F_{g,s}{}^{DY}(R_1)$;
- $T = pk_u F_{g,s}{}^{DY}(R_2)^r$;
- $VerifySig(pk_{TTP}, (sku, s), \sigma_{TTP}(sk_u, s)) = true$ ZKPOK in 4) is acquired in the following way:

a) P_U calculates $B = PedCom(sk_u)$, $C = PedCom(s)$; and proves the commitment values of B and C are CL signatures of TTP;
b) It should be proven that $S = F_{g,s}{}^{DY}(R_1) = g_2^{1/s+R_1+1}$ and $T = pk_u F_{g,s}{}^{DY}(R_2)^r = g_2^{sk_u + r/(R_2+s+1)}$, and the result of knowledge signature is ϕ.
5) P_M verifies the validity of $Cert_{rater} = (S, \pi = (r, T, \phi))$. If it is valid, P_M temporarily saves the rater certificate $Cert_{rater}$.

6.4 Evaluated certificate generation protocol

P_M generates evaluated certificate $Cert_{ratee}$ based on the information on its own pseudonym and its certificate container W. $Cert_{ratee}$ is generated in the same way as $Cert_{rater}$, $Cert_{ratee} = (V, \psi = (W, t, \omega))$ where:

$V = F_{g,s_1}{}^{DY}(R_1) = g_2^{1/s_1+R_1+1}$

$W = pk_u F_{g,s_1}{}^{DY}(R_2)^t = g_2^{sk_u + t/(s_1+R_2+1)}$, $R_1 = H(0 \Pi P_M)$, $R_2 = H(1 \Pi P_M)$. s_1 is the secret of evaluation certificate container of. t is randomly generated by P_M. ω is the non-interactive zero-knowledge signature.

6.5 Reputation evaluation verifying protocol

1) P_M submits Rater certificate $Cert_{rater}$ and Ratee certificate $Cert_{ratee}$ to TTP. TTP first verifies ϕ. If ϕ is not valid, TTP abandons $Cert_{rater} = (S, \pi = (r, T, \phi))$; if ϕ is valid, TTP temporarily saves $Cert_{rater}$ submitted by P_M.
2) TTP subsequently verifies ω. If ω is not valid, TTP abandons $Cert_{rater}$ and $Cert_{ratee}$. If it is valid, TTP needs to further verify in the following way: 1) whether S equals V; 2) whether S is the same with all the S in $Cert_{rater}$ being deposited. If the above two conditions are met, TTP accepts $Cert_{rater}$ and $Cert_{ratee}$, sends feedback message of reputation evaluation verified to P_M, and deposits $Cert_{rater}$ and $Cert_{ratee}$.

6.6 Reputation value updating protocol

After $Cert_{rater}$ and $Cert_{ratee}$ are verified, reputation value of P_M should be updated. But since the reputation account of P_M is bound to the real identity U_M of P_M. If

P_M updates reputation value directly, TTP will know P_M is the temporary certificate of U_M, and reveal the link between user's temporary certificate and real identity. Assume that U colludes with TTP. U will know P_M is the pseudonym of M, which poses grave threat to M's privacy. In order to solve this problem, blind signature is introduced when updating reputation value [28]. Two steps are adopted:

1) P_M chooses random information D, and sends it to TTP. TTP gives blind signature on D. P_M acquires blind permission σ. P_M deposits blind permission σ in its own database.

M sends the blind permission (D,σ) to TTP with its real identity U_M. TTP verifies whether the blind permission was used. If not, TTP updates the reputation value of M in reputation account.

6.7 Attacker identifying

6.7.1 Identifying the attacker

If the certificate identification S in $Cert_{rater}$ equals certificate identification V in $Cert_{ratee}$, it means two temporary certificates of the same user are making mutual evaluations. Since in $S = F_{g,s}{}^{DY}(R_1) = g_2^{1/s+R_1+1}$ and $V = F_{g,s_1}{}^{DY}(R_1) = g_2^{1/s_1+R_1+1}$, R_1 is the same, and the secret of each node s is unique, then s must be the same with s_1. TTP can calculate the user's public key through two double evaluation equations $T = pk_u F_{g,s}{}^{DY}(R_2)^r = g_2^{sk_u+r/(R_2+s+1)}$ and $W = pk_u F_{g,s_1}{}^{DY}(R_2)^t = g_2^{sk_u+t/(s_1+R_2+1)}$, and generate violation proof $\Pi_G = (\pi_1, \pi_2)$. The calculation of the user's public key is as follows:

$$pk = (\frac{T^t}{W^r})^{(t-r)^{-1}} \tag{3}$$

If S is the same with any S in $Cert_{rater}$ saved by TTP, it shows the frequency of mutual evaluations made between two users exceeds a limit. For Rater, $(r_1, T_1) \in \pi_1$, $(r_2, T_2) \in \pi_2$ have the same identification S. Through two double evaluation equations $T_1 = pk_u F_{g,s}{}^{DY}(R_2)^{r_1} = g_2^{sk_u+r_1/(R_2+s+1)}$ and $T_2 = pk_u F_{g,s}{}^{DY}(R_2)^{r_2} = g_2^{sk_u+r_2/(R_2+s+1)}$, TTP calculates the identity of Rater and generates the violation proof $\Pi_G = (\pi_1, \pi_2)$. The calculation of the public key of Rater is as follows:

$$pk_{rater} = (\frac{T_2^{r_1}}{T_1^{r_2}})^{(r_1-r_2)^{-1}} \tag{4}$$

For Ratee, $(t_1, W_1) \in \psi_1$, $(t_2, W_2) \in \psi_2$ have the same identification V. Through two double evaluation equations, $W_1 = pk_u F_{g,s_1}{}^{DY}(R_2)^{t_1} = g_2^{sk_u+t_1/(s_1+R_2+1)}$ and $W_2 = pk_u F_{g,s_1}{}^{DY}(R_2)^{t_2} = g_2^{sk_u+t_2/(s_1+R_2+1)}$, TTP calculates the identity of Ratee and generates the violation proof $\Pi_G = (\varphi_1, \varphi_2)$, The calculation of the public key of Ratee is as follows:

$$pk_{ratee} = (\frac{W_2^{t_1}}{W_1^{t_2}})^{(t_1-t_2)^{-1}} \tag{5}$$

6.7.2 Verifying violation proof

Violation proof $\Pi_G = (\pi_1, \pi_2)$, where $\pi_i = (r_i, T_i, \phi_i)$. Operate Identity(params, S, π_1, π_2) algorithm, compare the output with the output of public key pk_u. Use it as input and check if these two match. Subsequently, verify ϕ_i related to (S_i, r_i, T_i). If both are verified as valid, then VerifyGuilt(params, S, Π_G, pk_u) accepts the violation proof, otherwise it refuses the violation proof. TTP records the violator in the set Ψ.

6.8 Tracing malicious nodes

Complete anonymity does not mean that none of evaluations of deals can be traced. If that is the case, such a system will bring convenience to outlaws. This paper has the following considerations when tracing malicious nodes: a) if it is a valid evaluation, TTP cannot identify the user's real identity, or trace the user; b) if the violation of a node occurs less than d times in a limited amount of time, the node will not be traced; c) if the violation of a node occurs more than d times in a limited amount of time, the system will trace the violator. In order to trace a violator, TTP needs to acquire the key s of the violator's certificate container. The process is as follows:

6.8.1 The generation of a sub-key

By using Feldman's non-interactive verifiable key sharing, assume d is the pre-set threshold value. If the public key pk_u of the violator first appears in the set Ψ of TTP, the node needs to select a $d-1$ key sharing polynomial $f(x)$. The calculation of polynomial coefficient is: $k_i = 1/(s+R+1+i)$, where $R = H(pk_u)$, s is the secret of evaluation container, $i_i = 0,...,d$; the polynomial generated is $f(x) = k_0 + k_1 x + ... K_d x^d \bmod q$, $C_i = g^{k_i} \bmod p$. Take $\{C_0,...,C_d\}$ as the coefficient's commitment value of the polynomial $f(x)$. The sub-key $s_i = (pk_u, \{C_i\}_{i=0...d}, x, f(x), d)$, $x = V = H(r_i \,||\, T_i)$. d is the threshold value.

6.8.2 Verifying the sub-key

When the node updates its reputation value, if pk_u is in the violators set Ψ in the time limit $t(n)$, the peer must submit valid s_i. The number of s_i submitted depends on the times of the number of pk_u in Ψ this time. Otherwise,

TTP refuses to update the reputation value. The process of verifying the sub-key:

a) TTP verifies the received sub-key $(pk_u, \{C_i\}_{i=0...d}, x, f(x), d)$. Through calculating $g^{f(x)} = C_0 * C_1^x * ...C_d x^d \bmod p$, TTP verifies whether $(x, f(x))$ is the node of the polynomial $f(x)$ based on $\{C_i\}$ commitment. If not, TTP refuses the request of updating the node's reputation value.

b) If a sub-key with the public key being pk_u already exists, TTP needs to verify whether x was used, and whether $\{C_0, ..., C_d\}$ are the same。

When the certificate is verified as valid, TTP updates Ψ , and deletes pk_u from Ψ .

6.8.3 The leaking of automatic threshold

In the time period t, for the same evaluation public key pk_u , if TTP acquires d sub-keys s_i , TTP can acquire the key s of the certificate container by using the d sub-keys s_i , and reconstructing polynomial $f(x)$. The detailed process:

a) TTP reconstructs the polynomial $f(x)$ by using the d sub-keys s_i ;

Extract the constant term $f(0) = k_0 = 1/s + R + 1$ of the polynomial $f(x)$, where $R = H(pk_u)$. The key of the container s can be acquired.

6.8.4 Tracing malicious nodes

TTP uses $S = F_{g,s}{}^{DY}(R_1) = g_2^{1/s+R_1+1}$ to calculate the evaluation certificate with the node's public key being pk_i , and can trace all evaluation and transaction records of the node with the public key being pk_i .

7 Protocol analysis

7.1 Validity analysis

Theorem 1: A node cannot generate an invalid evaluation certificate through a certificate container.

Proof: Adversary A and the knowledge extractor that acts as TTP $K = K_{Withdraw_{m,L_S}}^{BB-K(A)}$ (knowing the private key of TTP) execute f Withdraw . Assume that the first step proof of knowledge (commitment C') of Withdraw protocol is interactive, and $K` = K_{Withdraw_{m,L_S}}^{BB-K(A)}$ is the extractor of knowledge proof. For the certificate withdrawal protocol Withdraw , except when K executes the code of K` in the first step to extract the value from (sk_u, s) , K is considered a truthful TTP. After acquiring secret s, and K can the calculation $\alpha = (S_1, S_2 ..... S_m, sk_u, s)$ be accomplished, where $S_i = F_{g,s}{}^{DY}(R_i)$ (R_i is the temporary information on Ratee's temporary certificate), $C' = PedCom(SK_u, s'; r)$. When the execution of *Withdraw* protocol ends, K outputs $(state_K, C', \alpha) \in L_S$, where C' is the commitment, and equals $PedCom(sK_u, s)$ the sent from adversary A in the Withdraw protocol. Language L_S is the triple $(., C', \alpha)$, where the first element can be of any value, but the second must be consistent with α . $A_f = (S_{j,\mathrm{i}} | \le j \le f, 1 \le i \le n)$ is the identification after f executes *Withdraw*.

If the knowledge proof in *Withdraw* protocol is non-interactive, except the black box A accesses, K and K' are both controlled by random oracle.

A_f includes all certificate tags of all deal-makers. For A to win the game, it must make sure that TTP will accept an evaluation certificate of a non-truthful proof with a relatively high probability.

Assume A is sure that a truthful TTP receives invalid identification S in reputation evaluation verifying protocol, where $S \notin A_f$. Then A must forge a wrong proof: a) A must know TTP's signature on public B, C; b) S and T will be generated in the proof Γ. Under the assumption that CL signature is secure (based on strong RAS assumption and LRSW), a) occurs with the probability that $V_1(k)$ is negligible; under the discrete assumption, b) occurs with the probability that $V_2(k)$ is negligible. Then A occurs with the probability of $V_1(k) + V_2(k)$. Thus, the total success probability of A is negligible.

7.2 Anonymity analysis

Theorem 2: The identity anonymity of Ratee and Rater should be ensured in executing the protocols.

Proof: Adversary A can act as a corrupted TTP and Ratee, generate and publicize the public key pk_{TTP} . Adversary A can execute *Withdraw* protocol for any times with any node. Eventually, adversary A will execute evaluation certificate generation protocol *RaterCert* with some real Raters (having certificate container W_j) or simulator S (without W_j).

Simulator $S = S^{IO-RO(A)}(Params, auxsim, H(P_{id}))$ gives the input: public parameter params, supplementary information auxsim and $H(P_{id})$ (P_{id} is the temporary certificate of Ratee). Given the random oracle model of the simulator and the input/output controlling model that adversary A accesses, simulator S executes evaluation certificate generation protocol *RaterCert* that adversary A requests in the following way:

a) Simulator S receives a character string info $\in \{0,1\}^*$;

b) Simulator S collects information on exchange, like info, pk_{TTP} , time, etc., the fixed output of S as a random oracle is a random value $r \in Z_q^*$;

c) S randomly chooses the secret s and calculates $S = F_{g,s}{}^{DY}(R_1)$ and $T = pk_u F_{g,s}{}^{DY}(R_2)^r$;

d) S sends a reputation evaluation certificate $Cert_{rater} = (S, \pi = (r, T, \phi))$ to A. ϕ is the simulated signature proof generated through the following steps:

- $B = PedCom(sk_u)$, $C = PedCom(s)$ and a simulated proof that proves the commitment values of B and C, which is based on CL signature of TTP; (in this step, S switches to CL simulator in random oracle model).
- A real proof $S = g^{1/s+R_1+1}$ and the proof of $T = g^{sk_u+r/(R_2+s+1)}\ \Gamma$.

All the difficult problems of simulator S are knowledge proof problems about processing CL signature through the simulator. When withdrawing the protocol, due to CL signature, A does not know the key s of the certificate container and the user's private key sk_u . Hence, the s simulator S chooses and the s the real user chooses are not mutually distinguishable. The components of evaluation certificate are $Cert_{rater} = (S, \pi = (r, T, \phi))$, where r is selected by adversary or RO model (which are not mutually distinguishable). Based on the security of DY PRF, the identification S in the evaluation certificate and the double evaluation equation T are not distinguishable from the random elements in G_2 . Since $F_{g,s}{}^{DY}(R_2)^r$ is calculated in cyclic groups of prime numbers, T looks random. Simulated ϕ consists of two real proofs and a simulated proof. The simulation proof is the only difference between the simulator S's proof ϕ and the real user's proof ϕ . Camenish Lysyanskaya simulator is operated to generate simulated proof. Due to the security of Camenish Lysyanskaya signature based on strong RSA or LRSW, the probability that A distinguishes Game R (from real user) and Game I (from simulator S) is negligible.

7.3 Identifying the attacker

Theorem 3: the malicious attacker will be identified.

Proof: Since r is randomly chosen by truthful Ratee M, the probability that r_1 equals r_2 is very low. With regard to each double evaluation equation $T_i = pk_u F_{g,s}{}^{DY}(R_2)^r$ that has the same pk_u and s, the direct use of the correct $Identity(\gamma, S, \pi_1, \pi_2)$ algorithm can successfully restore the violator's identity $pk_u = g^{sk_u}$.

Ratee M (or random oracle) chooses r_1 and r_2 . This determines the only $T_1 = pk_u F_{g,s}{}^{DY}(R_2)^{r_1} = g^{sk_u+r_1/(s+1)}$ $T_2 = pk_u F_{g,s}{}^{DY}(R_2)^{r_2} = g^{sk_u+r_2/(s+1)}$, which appears as the only valid and secure double evaluation equation with the evaluation identification S in the two reputation evaluation certificates. In order to generate different identifications in executing the evaluation certificate generation protocol $RaterCert$, A must change the valid proof π in the 4th step of the certificate generation protocol (in "validity proof", it has already been proven that the probability is negligible).

Identity algorithm can output the public key pk_u of the attacker, and the malicious Rater's proof $\Pi_G = (\pi_1, \pi_2)$, because VerifyGuilt accepts only truthful Identity output, and truthful public key pk_u and Π_G will be accepted by VerifyGuilt(params, S, Π_G, pk_u) with a relatively high probability.

7.4 Non-frameability analysis

Theorem 4: The forged evaluation certificate proof of other nodes can be identified.

Proof: Any valid proof π_i should concern the knowledge proof of the user's private key (refer to evaluation certificate generation protocol). If the two reputation evaluation certificates (S, π_1) and (S, π_2) of the violation proof $\prod_G = (\pi_1, \pi_2)$ are accepted by a truthful TTP, it means there are two possibilities: 1) adversary A must successfully forge the knowledge proof. In "validity proof", it has already been proven that the probability is negligible; 2) if (S, π_1) and (S, π_2) are accepted as different reputation evaluation certificates for they are registered as different users, VerifyGuilt algorithm will refuse $\Pi_G = (\pi_1, \pi_2)$, because Identity algorithm cannot restore the user's public key pk_u through the double evaluation equation generated by different public keys pk_u and secret s.

8 Comparison and contrast

This paper adopts similar schemes to [31] and [32]. This chapter mainly compares this scheme with that of [31] and [32] in efficiency, anonymity and security.

8.1 Efficiency comparison

Table 1 Efficiency comparison

Scheme	Calculation (modular exponent)		Complexity (Messages)
The scheme	Rater	11	$O(1)$
	TTP	16	
[31] scheme	Rater	27	$O(1)$
	TTP	16	
[32] scheme	Rater	more than 30	$O(1)$
	reputation value recipient	more than 30	

The calculation of this scheme, and those from [31] and [32] is mainly on multi-based modular exponent and single-based modular exponent. But through proper methods, calculations like $g_1^{x_1} g_2^{x_2}$ can be taken as 1,2 single modular exponent, and calculations like $g_1^{x_1} g_2^{x_2} g_3^{x_3}$ can be taken as 1,5 modular exponent.

Considering that the evaluation certificate container withdrawal protocol in this scheme, e-cash withdrawal protocol in [31] and user registration protocol in [32]can all be executed offline, the comparison on efficiency will mainly focus on the executing efficiency of these three

schemes in terms of reputation evaluation and reputation verifying protocol.

In this scheme, Rater needs to execute modular exponent calculations for 11 times to generate reputation evaluation certificate. TTP needs to execute modular exponent calculations for 16 times to verify the validity of the evaluation certificate and the Ratee certificate; in [31], Rater needs to execute modular exponent calculations for 27 times to generate e-cash that acts as evaluation value. In the reputation evaluation verifying protocol, TTP needs to execute modular exponent calculations for 16 times to verify the validity of e-cash; in [32], reputation Rater needs to execute modular exponent calculations for more than 30 times to generate the signature of reputation value, and the reputation value recipient also needs to execute modular exponent calculations for more than 30 times to verify the signature of the reputation value. Compared with [31] and [32], in terms of the complication of calculation, this scheme has made improvements.

In terms of the complication of information, the three schemes are basically the same.

8.2 Comparison of anonymity and security

Both this scheme and [32] adopt zero-knowledge proof and blind signature technology to realize the anonymity of the Ratee. Though [31] adopts e-cash technology, this technology is based on zero-knowledge proof. Thus, the privacy technology used in these three schemes is similar; all three realized the anonymity of Ratee. But in terms of security, this scheme can resist and identify Sybil attack and ballot-stuffing attack. The work in [31] cannot resist or identify any malicious attack. And [32] can only resist Sybil attack.

Table 2 Anonymity and security

Scheme	Key Security Mechanisms	Ratee's anonymity	Security
This scheme	Zero-knowledge proof, blind signature	supportive	Resisting and identifying Sybil and ballot-stuffing attack
[31] scheme	e-cash	supportive	Not being able to identify the attacker
[32] scheme	Zero-knowledge proof, blind signature	supportive	Resisting Sybil attack

9 Conclusion

This paper addresses such problems as the conflict between anonymity and reputation and the uncontrollability of reputation evaluation that are prominent in reputation system where nodes' identity is absolutely anonymous. It introduces conditional anonymity into reputation system to anonymously monitor the reputations evaluations made by users. Only when the anonymity condition is met can the reputation value be gained. Violators will be exposed and punished. Nodes that exceed the violation frequency in a certain time limit will be traced. Reputation evaluations made between users will be controlled. Based on blind signature, when a user updates the reputation value, its privacy can be protected, and its temporary certificates unlinked. It can also reconcile the conflict between anonymity and reputation.

10 Reference

[1] Jøsang, A.; Ismail, R.; Boyd, C. A survey of trust and reputation systems for online service provision. // Decision Support Systems. 43, 2(2007), pp. 618-644. DOI: 10.1016/j.dss.2005.05.019

[2] Seigneur, J.; Jensen, M. Trading Privacy for Trust. // The International Conference on Trust Management Proceedings of iTrust'04, Oxford, UK. vol. 2995, (2004), pp. 93-107. DOI: 10.1007/978-3-540-24747-0_8

[3] Gupta, M.; Judge, P.; Ammar, M. A reputation system for peer-to-peer networks. // NOSSDAV '03 Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video, New York, 2003, pp. 144-152.

[4] Aberer, K.; Despotovic, Z. Managing Trust in a Peer-2-Peer Information System. // In Proc. of the 10th International Conference on Information and Knowledge Management (CIKM01) / New York, 2001, pp. 310-317. DOI: 10.1145/502585.502638

[5] Steinbrecher, S. Enhancing multilateral security in and by reputation systems. // In FIDIS/IFIP Internet Security and Privacy Summer School. 2008, pp. 135-150.

[6] Pingel, F.; Steinbrecher, S. Multilateral secure cross-community reputation systems for internet communities. // Proceedings of the Fifth International Conference on Trust and Privacy in Digital Business (TrustBus2008) / Italy. Vol. 5185, (2008), pp. 69-78. DOI: 10.1007/978-3-540-85735-8_8

[7] Steinbrecher, S. Design options for privacy-respecting reputation systems within centralised internet communities. // Security and Privacy in Dynamic Environments, Karlstad, Sweden. Vol. 201, (2006), pp. 123-134. DOI: 10.1007/0-387-33406-8_11

[8] Damiani, E.; di Vimercati, S. D. C.; Paraboschi, S. et al. Managing and sharing servents' reputations in p2p systems. // IEEE Transaction on Knowledge and Data Engineering. 15, 4(2003), pp. 840-854. DOI: 10.1109/TKDE.2003.1209003

[9] Kamvar, S. D.; Schlosser, M. T.; Garcia-Molina, H. The eigentrust algorithm for reputation management in P2P networks. // Proc. of the 12th Int'l World Wide Web Conf. / Budapest. 2003, pp. 640-651.

[10] Kinateder, M.; Terdic, R.; Rothermel, K. Trusting pseudonyms–anonymous communication in Peer-to-Peer reputation systems. // International Journal for Infonomics. (2005), pp. 1-13.

[11] Ismail, R.; Boyd, C.; Jøsang, A. et al. A Security Architecture for Reputation Systems. // Proceedings of the 4th International Conference of E-Commerce and Web Technologies, Prague, Czech Republic. vol. 2738, (2003), pp. 176-185. DOI: 10.1007/978-3-540-45229-4_18

[12] Ismail, R.; Boyd, C.; Jøsang, A.; Russel, S. Private reputation shemes for P2P systems. // In WOSIS, 2004/2004, pp. 196-206.

[13] Androulaki, E.; Choi, S.; Bellovin, S. et al. Reputation Systems for Anonymous Networks. // Proceedings of the 8th International Symposium on Privacy Enhancing Technologies, Leuven, Belgium, vol. 5134, (2008), pp. 201-218. DOI: 10.1007/978-3-540-70630-4_13

[14] Santos, Q.; Hasan, O.; Brunie, L. A Distributed Privacy-Preserving Reputation Protocol Resistant to Reflection Attacks. // Rapport de recherche RR-LIRIS-2014-004, 2014.

[15] Petrlic, R.; Lutters, S.; Sorge, C. Privacy-Preserving Reputation Management. // In: Proceedings of the 29th Symposium on Applied Computing, Gyeongju, Korea. 2014, pp. 1712-1718. DOI: 10.1145/2554850.2554881
[16] Vasirani, M.; Wijaya, T. K.; Liassas, G. et al. Privacy Enhanced Demand Response with Reputation-based Incentive Distribution. // In: International Workshop on Demand Response, Cambridge, UK. 2014, pp. 1-6.
[17] Voss, M. Privacy preserving online reputation systems. // In International Information Security Workshops, Toulouse, France. vol. 148, (2004), pp. 245-260.
[18] Chaum, D.; Fiat, A.; Naor, M. Untraceable electronic cash. // In CRYPTO'90, Santa Barbara. California, USA. vol. 403, (1990), pp. 319-327.
[19] Camenisch, J.; Hohenberger, S.; Lysyanskaya, A. Compact E-Cash. // In EUROCRYPT '05. vol. 3494, (2005), pp. 302-321.
[20] Jarecki, S.; Lincoln, P.; Shmatikov, V. Negotiated Privacy - Software Security: Theories and Systems: Mext-NSF-JSPS International Symposium. // In: ISSS 2002, Tokyo, Japan. 2002, pp. 8-10.
[21] Jarecki, S.; Shmatikov, V. Handcuffing big brother: an abuse-resilient transaction escrow scheme. // In EUROCRYPT, Interlaken, Switzerland. vol. 3027, (2004), pp. 590-608.
[22] Nguyen, L.; Safavi-Naini, R. Dynamic k-times anonymous authentication. // In ACNS. vol. 3531, (2005), pp. 318-333.
[23] Camenisch, J.; Hohenberger, S.; Kohlweiss, M. et al. How to win the clonewars: efficient periodic n-times anonymous authentication. // In: 13th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 2006, pp. 201-210. DOI: 10.1145/1180405.1180431
[24] Fujisaki, E.; Okamoto, T. Statistical zero knowledge protocols to prove modular polynomial relations. //I n Advances in Cryptology CRYPTO '97, Santa Barbara, California. vol. 1294, (1997), pp. 16-30.
[25] Camenisch, J.; Lysyanskaya, A. A signature scheme with efficient protocols. // In: Security in Communication Networks '02, Amalfi, Italy. vol. 2576, (2002), pp. 268-289.
[26] Dodis, Y.; Yampolskiy, A. A Verifiable Random Function with Short Proofs Keys. // In Public Key Cryptography. Vol. 3386 of LNCS, (2005), pp. 416-431.
[27] Pedersen, T. P. Non-interactive and information-theoretic secure verifiable secret sharing. // In CRYPTO '92. vol. 576, (1992), pp. 129-140.
[28] Juels, A.; Luby, M.; Ostrovsky, R. Security of blind digital signatures. // In Advances in Cryptology CRYPTO '97, Santa Barbara, California. vol. 1294, (1997), pp. 150-164.
[29] Fiat, A.; Shamir, A. How to prove yourself: Practical solutions to identification and signature problems. // In Advances in Cryptology CRYPTO '86, Santa Barbara, California, USA. vol. 263, (1986), pp. 186-194.
[30] Camenisch, J.; Stadler, M. Efficient group signature schemes for large groups. // In CRYPTO '97, Santa Barbara, California, USA. vol. 1296, (1997), pp. 410-424.
[31] Androulaki, E.; Choi, S.; Bellovin, S. et al. Reputation Systems for Anonymous Networks. // Proceedings of the 8th International Symposium on Privacy Enhancing Technologies, Leuven, Belgium. vol. 5134, (2008), pp. 201-218. DOI: 10.1007/978-3-540-70630-4_13
[32] Bethencourt, J.; Shi, E.; Song, D. Signatures of Reputation: Towards Trust without Identity. // In: Sion, R. (ed.) FC 2010 / Heidelberg. 6052, (2010), pp. 400-407.

Authors' addresses

Keli Zhang, PhD Candidate
Information Security Center,
Beijing University of Posts and Telecommunications,
No 10, Xitucheng Road, Haidian District,
100876 Beijing, P. R. China
E-mail: kelicybergirl@163.com

Yi Xian Yang, Professor
Information Security Center,
Beijing University of Posts and Telecommunications,
No 10, Xitucheng Road, Haidian District,
100876 Beijing, P. R. China
E-mail: yangyixian@bupt.edu.cn