A Multi-hop Privacy-Preserving Reputation Scheme in Online Social Networks

Linke Guo*, Xiaoyan Zhu†, Chi Zhang* and Yuguang Fang*†
*Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA
†National Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China
Email: {blackglk@, xiaoyanzhu@, zhangchi@, fang@ece.}ufl.edu

Abstract—**Online Social Networks (OSNs) are becoming immensely popular nowadays, and they change the ways people think and live. In this paper, we propose a novel reputation system which allows users to find potential connections between unfamiliar people and obtains a relative objective globally meaningful reputation value. To some extent, our scheme provides a way to judge people in OSNs without real interactions, but based on the existing overall attitudes on particular people. Moreover, our scheme can protect the confidentially of the potential relationships in which no one is able to acquire the detailed connections between two end nodes. Contrary to those which publish each individual's reputation online, we treat the reputation value in our system as a private issue that has been carefully guaranteed in our scheme.**

Index Terms—**Reputation Value, Reputation Path, Structured Encryption, Homomorphic Encryption, Signature.**

I. INTRODUCTION

Online Social Networks (OSNs) such as Facebook and Twitter enable people to communicate with other people or to share their own profiles on their web pages. People can use any wired or wireless devices to obtain the OSNs' services, and are also happy to remotely chat with their friends, or share images and video clips with each other. Nowadays, the phenomenon of pervasive usage of OSNs has been studied among several unique academic areas, such as psychology, business, and social science. One of the interesting functionality of OSNs is the reputation system which provides a way to evaluate a merchant, a company or even one person. Furthermore, it has been prevalent implemented due to its intuitive explanation which directly reflects the level of QoS (Quality of Service) or the degree of trust. However, we also observe thousands of hundreds of people complain about the leaking privacy information from such kinds of OSNs, especially among the commercial interactions on eBay. According to the newest report from Facebook, it has already reached 500 million users and still greatly increase, half of the active users log on in any given day, all of which seems gratifying the services. Due to the open nature of OSNs, people are allowed to view friends' information and friend list as well as they are checking their local storage, and also that kind of information has been leaked to the service providers as well. As far as we know, existing OSNs providers do not well protect the relationship information, which means the relationship between friends might be potentially leaked, leading to severe problems particularly in e-commerce environment. Therefore, there is an urgent need to prevent relationship and reputation information breaches while maintaining the original functionality of existing OSNs.

Related Works: A through survey [1] on the reputation system for online service provision describes the current trends and development in this area. There are various way to define the reputation or trust value in literature [2]–[6]. Our scheme does not focus on the definition and the derivation of the reputation value, instead, we are along the line of securely transmitting and collecting the value. In spite of various model of trust or reputation, several papers discuss the implementation of trust model on different types of networks, which, to some extent, further develop this area. For example, Xiong *et al.* in [7] design a reputation-based trust supporting framework to help estimate the trustworthiness of peers in P2P online community. Srinivasan *et al.* give the brief summary of reputation system in the ad hoc and sensor networks [8]. On the other hand, Guha *et al.* [9] firstly incorporate the trust and distrust in a computational trust propagation setting, which formulates the transitivity of the trust propagation, and our work is based on part of their work. In [10], Zhang *et al.* first-time define a formal framework to formalize the trust-based routing protocols by using routing algebra in WANETs, which also considers the problem of transitivity of trust. The results show the possibility of implementing the reputation system in OSNs, and all of these papers indicate the promising future of trust or reputation systems.

Our Contributions: In this paper, we focus on setting up relationship between two unknown nodes in OSNs while their potential links will provide an objective level of reputation. We extend the existing OSNs, where the rating is only implemented between two directly connected nodes, to a multi-hop scenario which reflects the real situation in OSNs. Our scheme satisfies the anonymous and confidential requirements for the potential relationships and the reputations, all of which are hidden from the service providers.

II. PRELIMINARY AND SYSTEM MODEL

A. Preliminary

1) Bilinear Pairings: Bilinear pairing operations are performed on elliptic curves [11]. Let G_1 and G_2 be an additive group and a multiplicative group, respectively, of the same prime order q. 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 $\forall a, b \in Z_q^*$, where Z_q^* denotes the multiplicative group of Z_q, the integers modulo q. In particular, $Z_q^* = \{x \mid 1 \le x \le q-1\}$ since q is prime.
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 $e(P, Q), \forall P, Q \in G_1$.

Pairings are the basic operations in the identity-based cryptosystem used as the authentication backbone in our scheme.

2) Proxy Re-Encryption: Proxy Re-encryption (PRE) is a cryptographic primitive which allows a proxy to convert the existing ciphertext encrypted with Alice's public or secret key into another one that can be decrypt by Bob's private or secret key without learning the detail about the underlying plaintext. PRE has two classifications, one is asymmetric re-encryption where given the proxy re-encryption key, it can translate the ciphertext under public key pk_a into that under another public key pk_b, and vise versa; on the other hand, we apply the symmetric re-encryption scheme in our system, where apart from transforming ciphertext using public keys, we re-encrypt the message by using another domain's secret key which satisfies both the efficiency and security requirements for revoking members in OSNs.

B. *Definitions and Assumptions*

We further define several entities and terminologies that we use in our proposed system. Besides, we also make assumptions for the behaviors of entities and the underlying scheme.

1) Key Distribution Center: It is responsible for system setup and distributing public/private key pairs to the members in the system. It will work offline unless there is a new user needed to be granted the access right.

2) Central Storage: We define a Central Storage (CS) for storing the reputation relationships. CS will respond to the query for neighbor information and updating the data storage. However, we assume that the CS will not be responsible for storing the reputation values, which is the same as the pervasive social networking sites nowadays, e.g., Facebook, Twitter and so on. For security concerns, we would not allow the members to upload information in the plaintext form onto the CS; on the contrary, we apply the cryptographic primitives to store both the social relationship and node information in ciphertext form.

3) Reputation Value: We define the reputation value as asymmetric numeric value with $r \in [0, 1]$ between two members, where 0 denotes lowest reputation level which may destroy the whole reputation path, 1 represents the highest level with full trust, respectively. Also, we assume that the reputation value is transitive, and the value of multiple path is the product of hop value. Furthermore, the asymmetric value means the directional reputation values can be different according to different trust levels. We also assume that the reputation value towards other members is a private issue to each member, in which members along the reputation path should not be aware of the reputation value to themselves, meanwhile, they cannot tell the reputation value to a specific member one hop further from him. We need to clarify that we cannot monitor the change of one node's reputation value towards others. More specifically, it is his right to arbitrarily change his "ratings" to others in the sense that we cannot treat this as a misbehavior.

4) Anonymity: In our scheme, the reputation path should be discovered and maintained confidentially to every member in the system, also hide from the CS. Furthermore, the reputation value along with the path should be kept anonymously and confidentially from other members who only act as "relay" nodes in the social networks, in the sense that an intermediate node would learn nothing concerning both the ancestor and successor nodes one hop away from it. We cannot perform the anonymity to one's friends in our scheme, since there are many means to trace back to the sender of a packet, i.e. IP address. Therefore, we only provide the scheme which offers the anonymity to the intermediate nodes along one path to the nodes that at least one hop further from them.

C. *Security Objectives*

The main security objective of our proposed scheme is to guarantee the privacy in the neighbor search during the path discovery and secure transmission of reputation values. Members' privacy includes the anonymous reputation value towards other interacted members and hiding the reputation path from all intermediate members. Anonymous reputation value requires that no intermediate member could tell the reputation value to his/her non-interactive members. Hiding reputation path implies that every nodes along the reputation path cannot obtain the routing table of the whole path. The members' privacy requirement should be fulfilled regardless of access rights of entities.

D. *Adversary Model*

The adversary model defines the attackers and their possible attacks on our proposed scheme. The predefined CS is a semi-trusted authority which is curious but honest. It will work online for the response of launched queries, but will not launch the malicious attacks (e.g., denial-of-service attack). CS will store the relationship information of the underlying members in the system as a whole. We allow the CS to eagerly acquire the information for every query. The members in the social networks will tend to acquire the reputation value not only to themselves, but also between arbitrary two members. Gaining such values will help them to intentionally build a reputation path based on their willings, which may make the final reputation value subjective and invalid. Malicious members in the system will attempt to be someone that have directly interaction with the target member, in which they will pretend to be the most trustful member compared to any other members but actually they may even have no connections at all, so that more reputation paths would be directly built from these attackers, which makes the system untrustworthy. Also, passive eavesdroppers will attempt to intercept the transmitted packets on the fly. We prohibit the collusion attacks in our scheme.

III. System Design

In this section, we present the design details for our privacy-preserving reputation scheme for OSNs which includes two main procedures: reputation path discovery and anonymous reputation value transmission.

A. *Main Idea*

We assume our scheme as a relatively stable system in which members maintains a relatively stable reputation values. Based on the careful observation on the pervasive online social networks nowadays, the central server will keep the record of friend list and allow people to update at any given time. Similarly, our reputation path discovery scheme is run recursively, in which each node will update his/her friend list to the CS continuously. We first apply the structured encryption [12] together with the proxy re-encryption for both the enquiry of neighbor nodes in an encrypted manner and update friend list without disclosing the detail to the server. Apart from these operations, we also design the new user grant and user revocation in an efficient and secure way. Each user will keep a star-like structure which maintains his relationships with the center and his friends (along with the updated indirected friends). When there is a request in search of a particular node, one will check his structure to discover that node. If it is involved in the structure, the request packet will be processed and passed along the reputation path. Thus, we hide the detail of the specific path, in the sense that every intermediate node only knows the next hop and the final destination, which provides the path anonymity and confidentiality.

B. Reputation Path Discovery

We use Fig. 1 as an example to illustrate our proposed scheme. In this figure, S node denotes the source node who will initiate the reputation transmission, while D node is the destination in which S is looking for. Note that S and D have no direct relationship stored in the CS. Other nodes are the potential nodes which would build up the reputation path when the transmission procedure is initiated. The red solid line represents the distribution of token, while the blue solid line denotes the query by using the previous required token. After CS process the query, it will return a set of nodes to whom is launched the request. Then, the node which launched the request will update a new set of nodes as the green dash line shows.

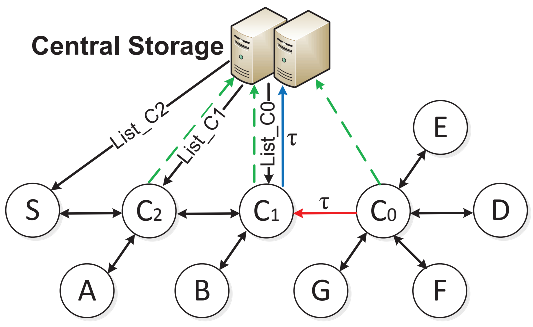

Fig. 1. Reputation Path Discovery.

1) System Setup: In our scheme, we apply the ID-based cryptographic technique for securing the reputation value transmission. In the system setup, the PKG will assign the ID-based public/private key pairs for each node in the system, and then PKG will go offline as we have assumed before. The key generation procedures are as follows [11]:

1. Input the security parameter ξ to the system and output a parameter tuple (q, G_1, G_2, e, P, H).
2. Randomly select a domain master secret $\varsigma \in Z_q^*$ and calculate the domain public key as $P_{pub} = \varsigma P$.

The PKG publishes the domain parameters tuple $(q, G_1, G_2, e, P, H, P_{pub})$ and maintain ς confidential, where $H(\cdot)$ is defined before as $H(\cdot) : \{0,1\}^* \to G_1$, and P 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 $H(ID)/\varsigma \cdot H(ID)$, which are distributed by the PKG during the initiation process. Each node will keep the private key secretly.

2) Data Uploading and Data Query: Now, we will introduce the basic uploading data scheme without exposing the data file in plaintext form, in which we call it anonymous storing. We utilize part of the structured encryption scheme proposed by Chase and Kamara in [12].

Initiation: Taking C_0 and C_1 as an example, C_1 will first generate a key tuple (α, β, κ) from a secret parameter k, where the process is operated by each possible device in the social network (laptop, PC, or a mobile device). We select two pseudo-random functions $\mathcal{F}$ and $\mathcal{G}$, and a semantic secure symmetric key encryption $\mathcal{E}$, and the key for the following scheme are α, β and κ, respectively:

$$\mathcal{F} : \{0,1\}^k \times V \to \{0,1\}^{max(L)\cdot\log\, n}$$
$$\mathcal{G} : \{0,1\}^k \times V \to \{0,1\}^k$$
$$\mathcal{E} : \{0,1\}^k \times \{0,1\}^l \to \{0,1\}^l$$

where $v \in V$ is the ID of vertex (node) in graph $G : (V, E)$, and $L(v_i)$ denotes the label associated to each node of the set $\{v_j \in V : (v_i, v_j) \in E\}$. Note that all the nodes (given the label $L(v_i)$) can be seen as starting from the node v_i, all of which form a star-like structure.

Based on the ID-based cryptosystem, we can also encrypt a message by using a symmetric key and the receiver can derive his symmetric key based on his local computation. Here, we assume K_{C_1,C_0}, and K_{C_0,C_1} represent symmetric keys generated by C_0 and C_1, respectively:

$$\begin{aligned} K_{C_1,C_0} &= e(H(ID_{C_0}), \varsigma H(ID_{C_1})) \\ &= e(H(ID_{C_0}), H(ID_{C_1}))^{\varsigma} \\ &= e(\varsigma H(ID_{C_0}), H(ID_{C_1})) = K_{C_0,C_1}. \end{aligned}$$

C_0 will transmit the symmetric key to its neighbors to decrypt the message obtained from CS as follows,

$$C_0 \to C_1 : E_{K_{C_0,C_1}}(\kappa), t_1, HMAC(E_{K_{C_0,C_1}}(\kappa)||t_1)$$

where t_1 is the timestamp designed to prevent reply attack and $HMAC$ is key hash message authentication code used for checking and verifying the integrity of the encrypted value. C_1 can acquire the decryption key for the packet generated by C_0.

Encryption: As we described before, we not only encrypt the data items on the node side, but also encrypt the data structure and stored in the CS. The data structure hidden from the server will disclose the corresponding nodes who have the authority to decrypt both the structure and the data file. We propose a efficient threshold structured encryption scheme which supports both data and user update.

We first define a random permutation $\pi : [n] \to [n]$, where $[n]$ is the total number of friends of a particular node. The friends or neighbors of v_i are stored in a sequential manner, which are represented as the element $m_\imath \in \mathbf{m}$, $1 \leqslant \imath \leqslant n$. Apparently, the element in $\mathbf{m}$ is the ID of v_i's friends in a plaintext manner. Consider the correlation between the plaintext of v_i's friends located in $\mathbf{m}$ and the location of their corresponding ciphertext, we define, $\mathbf{c}$. More precisely, let the random permutation π such that for a given $\imath \in [n]$, $m_\imath := \mathcal{D}_\kappa(c_{\pi(\imath)})$, where the $\mathcal{D}$ denotes the decryption process opposite to $\mathcal{E}$.

The encryption scheme includes two main parts: one is to encrypt the structure of the data items, the other is to encrypt the corresponding data set. In our scheme, nodes will upload a set of nodes undisclosed with a certain threshold, e.g., C_0 defines a threshold for his uploading policy in which $\delta_{C_0} = 0.6$, which means C_0 will store his neighbor nodes associated the reputation value greater than δ_{C_0}. We use term $\delta_{C_0}(L(C_0))$ to denotes the set of nodes in L_{C_0} that satisfy the requirement of reputation. Furthermore, this implies that any query based on token issued by C_0 will return the set of nodes with C_0's reputation value greater than 0.6. The proposed encryption is as follows, also taking C_0 as an example:

1. C_0 chooses a unique random permutation $\pi_{C_0} : [n] \to [n]$.
2. Compute the pseudo-random function by given key tuple β, let $kw := \mathcal{G}_\beta(C_0)$.
3. Store $(\pi_{C_0}(\imath)_{\imath \in \delta_{C_0}(L(C_0))}) \oplus \mathcal{F}_\alpha(C_0)$ onto the server with the search key kw, all of which form a encrypted structure T.
4. Permute $m_\imath \to m_\jmath$, where $\jmath \leftarrow \pi_{C_0}(\imath)$. For $1 \leqslant \jmath \leqslant n$, let $c_\jmath = \mathcal{E}_\kappa(m_\jmath)$.

Thus, the server will store a structure T and a set of corresponding ciphertext $\mathbf{c}$. We need to note that the entries stored in the server include all the encrypted pointers which concatenate with each other to XOR with $\mathcal{F}_\alpha(C_0)$, such that the maximum length of the results of $\mathcal{F}$ would be $max(L) \cdot \log\, n$. If there are not enough bits, we need to pad dummy bits to satisfy the length.

Decryption: Node C_0 will issue the query token to each of his friends whom C_0 consider it is certain, in the sense C_0 has the flexibility to choose whom to upload and issue the token. Decryption procedure is as follows,

1. Token issue: C_0 issues the token $\tau := (\mathcal{F}_\alpha(C_0), \mathcal{G}_\beta(C_0))$ to C_1 and a certain set of nodes.
2. Search: C_1 would be able to search by utilizing the token $kw = \mathcal{G}_\beta(C_0)$, and compute XOR for $\mathcal{F}_\alpha(C_0)$ and the given entry. If the given token is not in T, then output $\perp$; otherwise, the search result would be the pointers set $\mathcal{J} = (\jmath_1, ..., \jmath_n)$.
3. $\mathcal{J}$ points to the ciphertext which satisfies the token requirements and C_0's threshold. C_1 uses his symmetric key κ to decrypt the set of ciphertext that $\mathcal{J}$ points to, where $m_\jmath = \mathcal{D}_\kappa(c_\jmath)$, $1 \leqslant \jmath \leqslant n$.

3) Data and User Update: Our scheme requires each node in the system updates its own friend list recursively, which is used for the discovery of the reputation path. We also assume that the update process is run during the operation of the system.

Data Update: Data update is for the purpose of adding indirect friends to both nodes themselves and central storage. Taking C_1 as an example, first, it needs to update its own local path structure; second, it has to check whether the new updated node satisfy its own policy, e.g., if the C_0 upload the nodes with its reputation value greater than $\delta_{C_0} = 0.6$, $\delta_{C_1} = 0.5$ and $r(C_1, C_0) = 0.7$, C_1 needs to verify whether the node set obtained from C_0 satisfies δ_{C_1}; finally, C_1 rearranges its node set and updates to the server. We assume that every node will know its friends' threshold, e.g., C_1 will notice that $\delta_{C_0} = 0.6$. Thus, in this case, the reputation value range that C_1 towards C_0's uploaded nodes is between 0.42 and 0.7. Broadly speaking, there might have three possible conditions in large if comparing the own threshold and received reputation range,

$$\begin{cases} \Phi, & \delta_{C_1} \geqslant r(C_1, C_0) \\ \mathcal{S} \subseteq \mathcal{S}_\Omega^{\delta_{C_0}}, & \delta_{C_0} \cdot r(C_1, C_0) < \delta_{C_1} < r(C_1, C_0) \\ \mathcal{S}_\Omega^{\delta_{C_0}}, & \delta_{C_1} \leqslant \delta_{C_0} \cdot r(C_1, C_0) \end{cases}$$

where Φ denotes uploading nothing to the server and $\mathcal{S}$ is the subset of the whole set of the node C_0 uploaded to the server $\mathcal{S}_\Omega^{\delta_{C_0}}$. C_1 will upload the set of nodes to the server correspondingly according to their satisfaction toward the given reputation range. Now, we consider the most complicated condition where δ_{C_1} is between that two given values. However, as for C_1, it has little knowledge concerning which one or at most three of the received nodes are below its threshold. We design a negotiation process to handle this problem, which still achieves the security requirement for the system and we will discuss in later section. The negotiation process is as follows,

1. $C_1 \to C_0 : Ngt, t_2, E_{K_{C_1,C_0}}(\frac{\delta_{C_1}}{r(C_1,C_0)} || \mathcal{S}_\Omega^{\delta_{C_0}})$,
   $HMAC(Ngt||t_2||E_{K_{C_1,C_0}}(\frac{\delta_{C_1}}{r(C_1,C_0)} || \mathcal{S}_\Omega^{\delta_{C_0}}))$.
2. $C_0 \to C_1 : Rsp, t_3, E_{K_{C_0,C_1}}(\mathcal{S})$,
   $HMAC(Rsp||t_3||E_{K_{C_0,C_1}}(\mathcal{S}))$.

where Ngt and Rsp denote that the packets' purpose are for the negotiation of the uploaded nodes and response for that negotiation, respectively. Note that C_1 will transmit a quotient of $\frac{\delta_{C_1}}{r(C_1,C_0)}$ without disclosing the detail of the two values. After C_0 receives these packets from one of its friends, it will process and return the nodes $\mathcal{S} \subseteq \mathcal{S}_\Omega$ with its reputation values greater than that required quotient. Thus, C_1 can upload this set of nodes to the server which satisfy its own threshold δ_{C_1}.

We divide the data update process into two phases. The first phase is to rearrange the data sets and the following is to secure update file in the server along with the keyword. We first show the data rearrangement procedure in the local storage of C_1 as shown in Fig.2. Note that we exclude the node F because the C_0 deletes that node in response of C_1's negotiation request. If every node in the system uploads the corresponding nodes as above, it will hide the detail of the path from the node who asks for the token. If C_1 uploads this set of nodes as above, the node who requests the friend list from C_1 will know it can reach D via C_1 without knowing the detail who "relay" the packet to D.

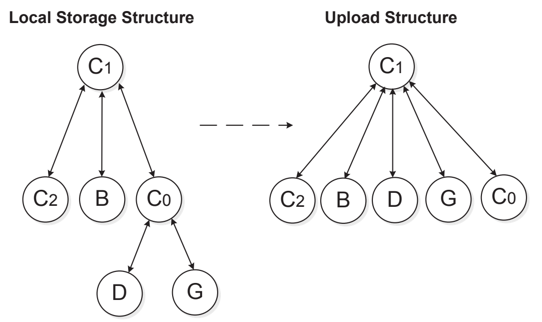

Fig. 2. Data Set Rearrangement.

During the phase of uploading updated data, C_1 needs to define a new random permutation π'_{C_1} and re-encrypt the message by using the original key κ_{C_1}. Then, C_1 needs to replace the original entry of the keyword with the new one $(\pi'_{C_1}(\imath)_{\imath \in \delta_{C_1}(L(C_1))}) \oplus \mathcal{F}_{\alpha_{C_1}}(C_1)$ without changing the token $\tau_{C_1} := (\mathcal{F}_{\alpha_{C_1}}(C_1), \mathcal{G}_{\beta_{C_1}}(C_1))$ and also the kw_{C_1}. Thus, the friends of C_1 can obtain the new set of friends by querying to the server by using the original token and decryption key, which saves the communication burden between C_1 and its friends.

User Update: We design the user update as part of our scheme to leverage the issue of new users grant and revocation of misbehaved users. We define the user who uploads data in terms of Data Owner. Compare to user revocation, the operation of new user grant is relatively simple and efficient. Since our scheme implement semantical symmetric key encryption for the nodes around data owner, it can directly issue the secret key κ and token to the new user if it satisfy the requirement of token policy (which is set by data owner). However, if the new user also needs to be updated to the server for other neighbors' search, the data owner can follow the algorithm of data updating to enlist that new user as well.

User revocation involves the key update and data re-encryption. Traditional ways incur extra computation and communication costs to the data owner. Since our scheme is based on the symmetric key encryption from end to the server, we only need upload newly re-encrypted data to the server and distribute the new symmetric key to the neighbors without including revoked users, which prevents the revoked users from decrypting the ciphertext using an original key. Thus, we propose to use proxy re-encryption scheme based on the assumption that CS is honest and will not launch malicious attacks (e.g., Denial of Service). Comparing to data update which has little effort on saving the communication costs but to re-upload one's friend list, or the search process will be messed up, the user revocation process needs to decrease the computation efforts on the user side. Taking

C_0 as an example, E is qualified to acquire the friend list from C_0, although E is not on C_0's uploaded friend list. C_0 may find that E behave badly which largely decreases C_0's reputation value and E is not able to query the friend list, so C_0 wants to revoke that user from obtaining the friend list. The process is as follows,

1. $C_0 \to CS : Rvk, t_4, E(SK^{\rho}_{C_0}), \sigma_{sk_{C_0}}(rvk||t_4||E(SK^{\rho}_{C_0}))$
2. CS : re-encrypt the ciphertext $\mathbf{c}_\jmath \to \mathbf{c}^{\rho}_\jmath := \mathcal{E}_{SK^{\rho}_{C_0}}(\mathbf{c}_\jmath)$, $1 \leqslant \jmath \leqslant n$.
3. $C_0 \to \mathcal{S}_{\Omega\setminus E} : Keyupd, t_5, E_{K_{C_0,\Omega\setminus E}}(SK^{\rho}_{C_0})$, $HMAC(Keyupd||t_5||E_{K_{C_0,\Omega\setminus E}}(SK^{\rho}_{C_0}))$.

where $SK^{\rho}_{C_0}$ represents the proxy re-encryption key that is used to encrypt the ciphertext, and $E(\cdot)$ denotes encryption function between CS and corresponding nodes. Here we clarify the different sets of nodes related to data owner, where $\mathcal{S}$ is a subset of v_i's uploaded nodes $\mathcal{S}^{\delta_{v_i}}_{\Omega}$ and $\mathcal{S}_{\Omega}(v_i)$ denotes all the possible nodes that are given the token to query for the data. Apparently, $\mathcal{S}^{\delta_{v_i}}_{\Omega} \subseteq \mathcal{S}_{\Omega}(v_i)$ denotes data owner will issue the token to the nodes which have been already uploaded to the server; on the other hand, it is the data owner's flexibility to choose to give the set $\mathcal{S}' \not\subset \mathcal{S}^{\delta_{v_i}}_{\Omega}$ but $\mathcal{S}' \subset \mathcal{S}_{\Omega}(v_i)$ the token. Specifically speaking, the user revocation process happen when the above condition holds. If a node has already uploaded and needs to be revoked, the data update and user revocation process should be operated accordingly. More generally, the policies based on different nodes will vary greatly, and we need to have fine-grained access control towards them.

C. *Secure Anonymous Reputation Value Transmission*

The reputation path algorithm helps us find the reputable path while maintaining the anonymity to the predecessors and successors, in the sense that the intermediate node will only relay the packets to the next node according to its own routing table without knowing the predecessors and successors. For example, when all the nodes in the system update their friend list according to our proposed scheme, S in Fig.1 will notice that it can find D via C_2 without knowing the detail of the path "after" C_2 if C_2 enlists D as its direct node. However, although S notices that D is on C_2's uploaded list and may know the threshold δ_{C_2}, it may not be proper to evaluate the objective reputation value toward D if δ_{C_2} is extremely lower than S's expectation. Therefore, we need to obtain an objective and accurate reputation value to evaluate that corresponding nodes in the system. We propose a secure reputation value transmission scheme which allows S to obtain an objective reputation value to D.

Before presenting our system setup, we need to clarify our design objective for the secure reputation value transmission. We keep the reputation value and reputation path as privacy issues which should be carefully protected, in the sense no intermediate nodes will notice the path and each one's reputation value in a disclosed form. However, keeping destination nodes confidentially from the intermediate node becomes meaningless since every interacted node needs to check its local storage (routing table) to find the destination. On the other hand, learning the destination nodes will not help the intermediate nodes find the path in detail. For example, C_1 should not know the path detail "before" C_2 whatever S is on C_1's local friend list, even if S is on its list, C_1 cannot distinguish whether it directly connects to C_2 or not, all of which fulfill the requirements of path confidentiallity and anonymity.

1) System Setup: Our construction makes use of an IK-CPA (*Indistinguishable Key under Chosen Plaintext Attacks*) encryption scheme which offers a multiplicative homomorphism. Informally, the key privacy property ensures it is infeasible to match a ciphertext with the public key used to produce it; this property is used to achieve path anonymity, which means attackers cannot distinguish the packets generated for different paths. Below, we give our key generation and distribution process based on IK-CPA secure scheme for the nodes on the existing reputation path.

1. ***Setup:*** Input the system with a security parameter λ and output a set of parameters, where $\Gamma = (f, \hbar) \in G_1$ and $f, \hbar$ are the generators of G_1, in which every node on a specific path will be aware of the path parameter Γ.
2. ***Key Geneneation:*** To generate a public/private key pair for a specific path, S will select a secret pair (a, b) as its private key (usk), where $a, b \in Z_p$, and compute the corresponding public key $upk := (f^a, \hbar^b) \in G_1$.

Note that S will generate different upk for different paths if there are multiple potential choices for S, in order to get more objective reputation value as a whole.

2) Anonymous Reputation Value Transmission: As mentioned before, the value of reputation is kept as a decimal between 0 and 1. However, the message space in our constructed encryption scheme is on a bilinear group G_1. Hence, we need to map the reputation value $r \in [0, 1]$ to our desired group, in the sense that we define a mapping function $U(\cdot) : r \in [0, 1] \to \{0, 1\}^{\ell} \in G_1$, where ℓ is a fixed length of any given element in the group. We also further assume that the results (elements) in multiplicative group G_1 satisfy the property of computebility, especially, multiplicativeness. After collecting the accumulative reputation value, any arbitrary node is able to find the original reputation value by applying the inverse function $U^{-1}(\cdot)$. For simplicity, we will still use r to represent the reputation value, where now $r \in G_1$. Therefore, we have the following interactions where the node-to-node secure communications are ensured, and we briefly write r_1, r_2, r_3 and r_4 instead of $r(S, C_2)$, $r(C_2, C_1)$, $r(C_1, C_0)$ and $r(C_1, D)$ for simplicity. Note that, as for S, it will not be aware of this path $S \to C_2 \to C_1 \to C_0 \to D$, but we use this path as an example to illustrate the scheme from outside of the system. We now construct the encryption and decryption scheme for the reputation value:

Encryption: To encrypt an arbitrary reputation $r \in G_1$ under the public key upk :=$(f^a, \hbar^b)$, we select two random exponents $\mu, \nu \in Z_p$ and compute the ciphertext as:

$$E_{upk}(r) = \mathbb{C} := < r \cdot (f^a)^{\mu} \cdot (\hbar^b)^{\nu}, f^{\mu}, \hbar^{\nu} > = < \mathbb{C}_1, \mathbb{C}_2, \mathbb{C}_3 > .$$

Decryption: For the source node, it will decrypt all the accumulated reputation values on different paths. By using the private key (a, b), the decryption algorithm is executed as follows:

$$\mathbb{C}_1 \cdot \mathbb{C}_2^{-a} \cdot \mathbb{C}_3^{-b} = \frac{r \cdot (f^a)^{\mu} \cdot (\hbar^b)^{\nu}}{(f^{\mu})^a \cdot (\hbar^{\nu})^b} = \frac{r \cdot f^{a\mu} \cdot \hbar^{b\nu}}{f^{a\mu} \cdot \hbar^{b\nu}} = r.$$

Accordingly, the transmission procedure is as follows,

1. $S \to C_2 : RPtran, seq, (E_{pk_D}(S), D), E_{upk}(r_1), t_6$, $SIG_{sk_S}(E_{upk}(r_1)||t_6)$;
2. $C_2 \to C_1 : RPtran, seq, (E_{pk_D}(S), D)$, $E_{upk}(r_2) \cdot E_{upk}(r_1)$, $t_7, SIG_{sk_{C_2}}(E_{upk}(r_2) \cdot E_{upk}(r_1)||t_7)$;
3. $C_1 \to C_0 : RPtran, seq, (E_{pk_D}(S), D)$, $\prod_{\iota=1}^{3} E_{upk}(r_\iota), t_8, SIG_{sk_{C_1}}(\prod_{\iota=1}^{3} E_{upk}(r_\iota)||t_8)$
4. $C_0 \to D : RPtran, seq, (E_{pk_D}(S), D)$, $\prod_{\iota=1}^{4} E_{upk}(r_\iota), t_9, SIG_{sk_{C_0}}(\prod_{\iota=1}^{4} E_{upk}(r_\iota)||t_9)$

where $RPtran$ not only denotes that this packet is for the purpose of transmitting reputation value, but also includes public parameters for this specific path (upk, Γ), and seq is an unique binary string represents each reputation path. In $(E_{pk_D}(S), D)$, note that D is able to decrypt and obtain the source ID S, and D is for the use of checking each node's routing table in its local storage. Note that each packet is encrypted under the symmetric encryption described before, in which all the packets are transmitted in a secure manner. Each node will encrypt the reputation value on the next node and multiply it with received encrypted value, then it will generate a signature for that product and transmit it to the following node according to its local storage.

3) Secure Reputation Value Response: We take the previous example to illustrate the scheme as well. when D receives:

$$\begin{aligned}\mathbb{C}_{total} &= E_{upk}(r_1) \cdot E_{upk}(r_2) \cdot E_{upk}(r_3) \cdot E_{upk}(r_4) \\ &= < \prod_{\iota=1}^{4} \mathbb{C}_{r_\iota,1}, \prod_{\iota=1}^{4} \mathbb{C}_{r_\iota,2}, \prod_{\iota=1}^{4} \mathbb{C}_{r_\iota,3} > \\ &= < \prod_{\iota=1}^{4} r_\iota \cdot (f^a)^{\mu_\iota} \cdot (\hbar^b)^{\nu_\iota}, \prod_{\iota=1}^{4} f^{\mu_\iota}, \prod_{\iota=1}^{4} \hbar^{\nu_\iota} >\end{aligned}$$

D will first check the field $(E_{pk_D}(S), D)$ and know that the destination node is itself. Then, it needs to prepare a returning packet letting S know the verifiable accumulative reputation value. We apply the an efficient ID-based Signature [13] based on pairing, which is based on the assumption that weak Diffie-Hellman problem is hard.

$D \to S :< E_{pk_S}(\mathbb{C}_{total}), t_{10}, SIG_{sk_D}(E_{pk_S}(\mathbb{C}_{total})||t_{10}) >$

Note that the returned packet is encrypted by $K_{D,S}$ and HMAC is used to protect non-repudiation. Then S will decrypt the message by using the private key $usk := (a, b)$ in the following way:

$$\begin{aligned}R_\omega &= \mathbb{C}_{total,\omega,1} \cdot \mathbb{C}_{total,\omega,2}^{-a} \cdot \mathbb{C}_{total,\omega,3}^{-b} \\ &= \frac{\prod_{\iota=1}^{4} r_{\omega,\iota} \cdot (f^a)^{\mu_\iota} \cdot (\hbar^b)^{\nu_\iota}}{(f^{\sum_{\iota=1}^{4}\mu_\iota})^a \cdot (\hbar^{\sum_{\iota=1}^{4}\nu_\iota})^b} \\ &= \prod_{\iota=1}^{4} r_{\omega,\iota} \cdot \frac{f^{a\cdot\sum_{\iota=1}^{4}\mu_\iota} \cdot \hbar^{b\cdot\sum_{\iota=1}^{4}\nu_\iota}}{(f^{\sum_{\iota=1}^{4}\mu_\iota})^a \cdot (\hbar^{\sum_{\iota=1}^{4}\nu_\iota})^b} = \prod_{\iota=1}^{4} r_{\omega,\iota}.\end{aligned}$$

S can derive the accumulative reputation value, and apply the inverse function $U^{-1}(R_\omega)$ to obtain the real decimal reputation value $\widehat{R}_\omega$. Let $\hat{\Omega}$ be the set of reputation values from which reputation values of paths have returned the corresponding values. Then, S can generate a relative reputation value for this specific node D:

$$\widetilde{R} = \frac{\sum_{\omega\in\hat{\Omega}} \widehat{R}_\omega}{|\hat{\Omega}|},$$

where the $|\hat{\Omega}|$ represents the cardinality of the set $\hat{\Omega}$ and $\omega \in \hat{\Omega}$ denotes each unique reputation path in $\hat{\Omega}$.

Accordingly, S finally acquires the global reputation value on a node that it has never met before. Based on such value, S can perform any possible interactions, e.g. commercial transactions, with D if the value satisfies the demand of S.

IV. Security Analysis

We briefly discuss the security objectives that our proposed scheme has achieved. Our scheme preserves the data confidentially in both the friend list query and anonymous reputation transmission processes, because our encryption schemes that used in uploading encrypted data, data queries and reputation value encryption are CPA-secure. To some extent, the ciphertexts which the scheme outputs do not reveal any partial information about the corresponding plaintexts even to an adversary who can adaptively query an encryption oracle. Another possible attack concerning data confidentiality is launched by attackers who intercept the packets transmission on the fly. However, all the encryption schemes we used during the transmission between two nodes are based on the assumption that Bilinear Diffie-Hellman Problem is hard. Our scheme also satisfies the security requirements of authentication and access control since nodes need to bilaterally verify each other by using each other's key pair to derive the symmetric keys. On the aspect of preserving the anonymous path, our scheme will not disclose the path information to any party in the system, since each node only keeps its next node as the records and uploads indirect nodes to the CS without exposing detailed paths. As a result, nodes cannot tell the previous and following nodes one hop away from them which perfectly protects anonymous reputation paths.

V. Conclusion

In this paper, we propose a privacy-preserving reputation scheme for online social networks, which securely find a reputation path between a pair of unknown nodes and also one can tell the globally reputation value on another one by implementing anonymous reputation value transmission procedure. Our scheme implements structured encryption with data update to form the reputation path to each node, meanwhile, it perfectly hide the detail routing information from the intermediate nodes along one path. Also, we treat one's reputation value on other nodes as a privacy issue which has been carefully treated and our scheme can let the intermediate nodes pass the reputation value without disclosing each hop's values, but the end user can find an objective reputation value in a disclosed form.

References

[1] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decis. Support Syst.*, vol. 43, pp. 618–644, March 2007.

[2] Y. Sun, Z. Han, and K. J. Ray Liu, "Information theoretic framework of trust modeling and evaluation for ad hoc networks," *IEEE Journal on Selected Area in Communications*, vol. 24, pp. 305–317, 2006.

[3] G. Theodorakopoulos and J.S. Baras, "On trust models and trust evaluation metrics for ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 318 – 328, Feb. 2006.

[4] A. Jøsang, "An algebra for assessing trust in certification chains," *Proceedings of the Network and Distributed Systems Security Symposium (NDSS'99). The Internet Society*, 1999.

[5] S. Capkun, L. Buttyan, and J.-P. Hubaux, "Self-organized public-key management for mobile ad hoc networks," *Mobile Computing, IEEE Transactions on*, vol. 2, no. 1, pp. 52 – 64, Jan. 2003.

[6] C. Zhang, Y. Song, and Y. Fang, "Modeling secure connectivity of self-organized wireless ad hoc networks," *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pp. 251 –255, Apr. 2008.

[7] L. Xiong and L. Liu, "Peertrust: supporting reputation-based trust for peer-to-peer electronic communities," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 7, pp. 843 – 857, July 2004.

[8] A. Srinivasan, J. Teitelbaum, J. Wu, M. Cardei, and H. Liang, "Reputation-and-trust-based systems for ad hoc networks," *Algorithms and Protocols for Wireless, Mobile Ad Hoc Networks, Wiley*, 2008.

[9] R. Guha, R.Kumar, P. Raghavan, and A. Tomkins, "Propagation of trust and distrust," *Proceedings of the 13th international conference on World Wide Web*, pp. 403–412, 2004.

[10] C. Zhang, X. Zhu, Y. Song, and Y. Fang, "A formal study of trust-based routing in wireless ad hoc networks," *INFOCOM, 2010 Proceedings IEEE*, pp. 1 –9, Mar. 2010.

[11] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," *Advances in Cryptology —CRYPTO 2001*, pp. 213–229, 2001.

[12] M. Chase and S. Kamara, "Structured encryption and controlled disclosure," *ASIACRPTO '10*, vol. 6477, pp. 581, 2010.

[13] F. Hess, "Efficient identity based signature schemes based on pairings," *Selected Areas in Cryptography*, pp. 310–324, 2003.