Abstract—Making new connections is a crucial service in current online social networks. However, such a service can also raise a big security concern. For example, two strangers become friends in the online social network and they want to communicate securely. The messages transmitted between them could be encrypted using their public keys and decrypted using their private keys. However, one can not determine whether this public key indeed belongs to the claimed user. In this paper, we design a system to authenticate two strangers in fast mixing online social networks. We make a thorough analysis on our system and show that there is a high probability of finding witness users in fast mixing social networks.

Index Terms—Authentication, Online Social Networks, Fast Mixing, Zero-Knowledge Proof

I. INTRODUCTION

Online Social Networks (OSNs), like Facebook, Twitter and Flickr, often suffer from impersonation attack, in which the adversary creates an OSN account using the profile of someone else other than himself and attempts to gain trust or benefit from the victims by impersonation. Recently, some researchers have shown that such attack is surprisingly effective and easy to launch[1].

An effective authentication system is necessary to defend against impersonation attack. For example, two people, say Alice and Bob, become friends in an OSN and Bob wants to securely send a message to Alice. He can encrypt the message using Alice’s public key and the asymmetric cryptographic system guarantees that only Alice can decrypt this message using her private key. Such a system can work if and only if the public key used by Bob in the encryption is indeed generated and exclusively owned by Alice. But how does Bob know this? Simply using a second secure communication channel, like email or cellular network, to transmit the public key is not sufficient because the two people involved probably do not know each other before. To protect users’ security and privacy in current OSNs, the most common approach is to deploy cryptographic systems, in which the authentication is to check if some cryptographic key indeed belongs to a specific identity.

One possible solution is to use a trusted centre, where all entities post their own public keys and request other people’s public keys via a secure channel. However, such a trusted centre, which suffers from the single point of failure, could become the bottleneck in a fast growing OSN, where people make friends and communicate with each other frequently.

It is challenging to design a decentralized authentication system in OSNs. However, we observe that an OSN is a super set of the real life social network, in which people have a strong connection with their 1-hop friends. In the real social network, a person could find a way to authenticate his friends, by either making a phone call or sending an email. We try to exploit this 1-hop trust to design a system that people can rely on to authenticate someone he never met before.

We assume that the trust in the real social network is transitive due to the social pressure. For example, if A is a neighbour of B and B is a neighbour of C. If A trusts B and B trusts C, then it is reasonable for A to trust C. 1-hop trust can be extended to multi-hop trust, i.e., a person could find a trusted person multiple hops away from himself in the social network. The basic idea of our system is that when two strangers meet in an OSN, if they can find the same trusted person, then they can rely on this common trusted person to authenticate each other.

A. Model

In our system, all communications take place in an OSN, which can be viewed as an undirected graph $G = (V,E)$. The vertices in the graph denote the users in the OSN and the edges denote the pair-wise relationships. Each user $u$ in the OSN has a unique identity $id_u$. A public/private key pair $(pk, sk)$ is independently generated by each user and $sk$ is kept as a secret. The basic assumption in our construction is that there exists some secure communication channel between any pair of 1-hop friends. The secure communication channel is such a channel that the two ends of this channel are authenticated by each other and the messages transferred through this channel is confidential. This kind of channel can be set up by any authentication and key exchange protocol[9]. However, in the real social network, the blue tooth, encrypted phone call or email can be viewed as a kind of secure channel because people usually know their close friend’s exclusive email address or phone number.

We assume that current social networks are fast mixing, which has been shown to be true in Yu’s recent work [20]. Intuitively, the property of “fast mixing” means the graph $G$ of the social network does not have small quotient cuts (whose removal will disconnect the graph). In other words, the nodes in the fast mixing networks connect other nodes evenly. A more specific explanation of fast mixing is given in Section II-A.

The adversary $A$ in our model is able to eavesdrop the communication channel between any two honest users, block, temper, replay or deliver out-of-order honest users’ messages and inject his own messages into the communication.

B. Security Goals

Our authentication system should achieve the following goals:

- an honest user is convinced that some public key $pk_u$ is attached to an identity $id_u$ if and only if the corresponding key pair $(pk_u, sk_u)$ is exclusively owned by the user $u$. 
• an adversary $A$ with a public key $pk_A$ is unable to convince an honest user that $pk_A$ is owned by another honest user $id_u$, even if $pk_A = pk_u$.

• the system is performed in a decentralized and asynchronous fashion. It is not necessary to require all users staying online during the execution of the protocols.

C. Overview of Our System

Our system leverages a pivot insight into the centralized authentication system (CAS). In a CAS, the entities query a digitally signed certificate binding their public keys with their identities from the certification authority (CA), i.e., trusted center. Then, they present the certificates to anyone who questions their ownership of the public keys. Along the same direction, we also utilize the concept of certificate in our system. In general, our system consists of two stages, the precomputed building-up stage and the online verification stage. In the building-up stage, the users query the certificates from a set of chosen users (witness users) instead of a specific CA. Simply, a certificate is the witness user’s digital signature on one’s public key and identity. At the end of the building-up stage, the users store all certificates along with the public keys of the witness users.

In the verification stage, one user asks another user to prove the ownership of the public key. The user who initiates the proof is called the authenticator. The other user is called the authenticatee. The two users first run a set intersection protocol to find a shared witness user $p$. Then the authenticator shows that he holds a certificate signed by $p$ via a zero-knowledge proof. Remarkably, the zero-knowledge proof is crucial to our system because in this way one can convince others to believe that he does own this certificate without leaking any information regarding the certificate. Hence, by running our protocol, a malicious user is unable to obtain an honest user’s certificate and replay it to other victims.

D. Related Work

The assumption that the social network is a fast mixing network was first proposed by Yu et al. in their SybilGuard system [21]. Later on, Yu et al. experimentally showed that this assumption holds true for OSNs in [20]. They used this characteristic structure to defend sybil attack in the social network because the sybil nodes hurt this feature by forming a small quotient cut in the network graph. When performing a random walk in an honest area, or a random routing as in [20], the probability that the walk escaped the honest area was bounded and hence the number of accepted sybil nodes was limited. Based on the same assumption, Danezis and Mittal proposed another sybil defence system[7]. After performing several random walks in the social network, they defined a probabilistic model and then used the Bayes theorem to infer the probability that a set of nodes were honest. Recently, Viswanath et al. [19] showed that four of the current sybil defence systems, including the above three ones, were equivalent to detecting communities in the social network. Actually, they used an existing community detection algorithm to achieve the similar experimental accuracy. Different from their work, we do not use the random walk itself to detect any community in our system but exploit the stationary distribution achieved by the random walk to guarantee the high probability of successful execution of our protocol.

Authentication has been extensively studied in various networks, including classic computer network, IP-network [13] [11], wireless sensor network[16][8] and cellular network[18]. Recently, Groe and Kat’s proposed a novel framework for authenticated key exchange between two parties based on a low entropy password in [9]. To our best knowledge, our paper is the first paper specifically discussing the authentication in OSNs.

The rest of this paper is structured as follows. We review necessary background and construct a new zero-knowledge proof of the Hohenberger-Waters signature [10] in section II. The details of our system construction are presented in section III. We show the simulation results and the conclusion in section IV and V respectively.

II. BACKGROUND

A. Random Walk and Fast Mixing

A random walk is a route on a graph. In a random walk, at each hop, the node in the graph randomly chooses an outgoing edge from all edges connecting to it. The walk is allowed to be directed back to the incoming edge. Random walk has been extensively studied in the random graph and complex network[15]. It is well-known that when performing a random walk in a connected and non-bipartite graph, the distribution of the last traversed node or edge gets to be independent of the starting node as the walk length goes to infinity[21]. This distribution is called stationary distribution. Mixing time is the time it takes for the random walk to achieve a state such that the distribution of the last arrived node or edge changes within some small variation distance to the stationary distribution [20]. Variation distance is to measure the difference between two distributions. We say a graph $G$ is fast mixing if its mixing time is $O(\log |V| + \log \frac{1}{\Delta})$, where $\Delta$ is the variation distance. In this paper we choose $\Delta = \frac{1}{|V|}$. Then we can simply say that a fast mixing graph is the graph with mixing time of $O(\log |V|)$. Yu et al. gave an important theorem about the fast mixing graph in [20] and we restate here:

**Theorem II.1.** Consider any fast mixing graph with $V$ nodes. A random walk of length $\Theta(\log |V|)$ is sufficiently long such that with probability at least $1 - \frac{1}{|V|^2}$, the last node/edge traversed is drawn from the node/edge stationary distribution of the graph.

B. Certificate Signature

We use the Hohenberger-Waters (HW) signature algorithm [10] to sign a certificate in our system. The HW signature is a stateful short signature in the standard model. Bilinear groups $\mathbb{G}, \mathbb{G}_T$ of prime order $p$ are used in the signature algorithm. A bilinear map from $\mathbb{G}$ to $\mathbb{G}_T$ is a function $e: \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ such that

- $e(g^a, g^b) = e(g, g)^{ab}$ for all $a, b \in \mathbb{Z}_p$;
- $e(g, g) \neq 1$;
- It can be computed efficiently.

where $g$ is a generator of the group $\mathbb{G}$. The verifying key in the HW signature algorithm is of the form $(g, g^a, u, v, d, w, z, h)$, where $a \in \mathbb{Z}_p$ and $u, v, d, w, z$ and $h$ are random generators of the group $\mathbb{G}$. The signing key is $a$. The signer locally maintains
a counter $s$. To sign a message $M$, one first increases $s$ by 1, selects two random numbers $r, t \in \mathbb{Z}_p$ and calculates the signature $(\sigma_1, \sigma_2, r, s)$, where $\sigma_1 = (w^M v^t d^o (w^{|\log(s)| + \#h})^t$ and $\sigma_2 = g^t$. In the verification, one checks if $e(\sigma_1, g) = e(g^a, w^M v^t d^o \cdot e(\sigma_2, w^{|\log(s)| + \#h})^t$ holds.

C. Zero-Knowledge Proof

Zero-knowledge proof of knowledge is to let others verify the proof of knowledge in a zero knowledge way, i.e., the proof leaks nothing about the knowledge itself. With the help of zero-knowledge proof, we are able to prove 1) a discrete logarithm modulo a prime[17], 2) the equality of some discrete logarithms[5] and 3) the conjunction of the previous two[6] in this paper. We follow the notation introduced and formally defined in [3] and [4] to denote the zero-knowledge proof. For example, $PK\{\alpha, \beta, \gamma : y = g^\alpha g_2^\beta \wedge z = b_1^2 h_2^3\}$ denotes a “zero-knowledge proof of integers $\alpha, \beta, \gamma$ such that the equation holds”. Here, $y, g, g_2, b_1, h_2 \in H, G = \{g_1\} = \{g_2\}$ and $H = \{h_3\} = \{h_2\}$.

Given this notation, one can easily derive a three-round protocol ($\Sigma$-protocol) to realize the proof. We refer to the literature [3] for the details of the derivation. Furthermore, the computation and communication overhead can also be derived from this notation[2].

We construct a zero-knowledge proof of the HW signature. Assume we are given a HW signature $(\sigma_1, \sigma_2, r, s)$ on a message $M$ and need to prove we indeed hold such a signature. First we choose $w_1, w_2, w_3 \in \mathbb{Z}_p$ in which the symbol $\mathbb{Z}_p$ denotes the previous number is chosen randomly from the group, compute $\tilde{\sigma}_1 = \sigma_1 w_1^3$, $\tilde{\sigma}_2 = \sigma_2 w_2^2$ and execute the following proof of knowledge

$PK\{r, s, |\log s|, x_1, x_2, x_3, \alpha, \beta : B = w_3^{x_2} w_2^{x_3} \wedge 1 = B^{-|\log s|} w_3^\alpha w_2^\gamma \wedge 1 = B^{-s} w_3^{x_2} w_2^{x_3}$
$\wedge e(\tilde{\sigma}_1, g) e(w_3, w)^\alpha e(w_2, h)^\gamma e(w_2, z)^\beta = e(g^\alpha, w)^M$
$e(g^a, v)^\gamma e(g^a, d) e(\tilde{\sigma}_2, w)^{log s} e(\tilde{\sigma}_2, z)^e(\tilde{\sigma}_2, h) e(w_1, g)^s\}$

where $\alpha = |\log s| x_2, \beta = s x_2, \gamma_1 = |\log s| x_3$ and $\gamma_2 = s x_3$. Note that all the bases are either public elements, like $w_2$ or $g^a$, or signature elements marked by random group elements, i.e., $\tilde{\sigma}_1, \tilde{\sigma}_2$. This is necessary because bases need to be exposed in the $\Sigma$-protocol and none of the above bases leaks the signature information.

III. System Constructions

We now describe our system in detail. Building-up stage is performed by each user in the OSN when he joins the network. Verification stage is invoked when two strangers want to authenticate each other.

A. Building-up Stage

In this stage, every user, which is called the initiator, samples a set of witness users by running the finding witness protocol for $m$ rounds. The construction of the protocol is shown in Figure 1.

Each round of the protocol performs a random walk in the OSN transmitting the initiator’s request and the witness user’s signature back along this walk (Figure 2(a)). This signature is a certificate binding the initiator’s identity and public key together. The random walk is of the length “$w$”. To achieve the stationary distribution, $w$ is required to be $O(|\log V|)$. Therefore, after $m$ rounds, the user is able to sample $m$ witness users and obtain their certificates. We denote user $u$’s witness user set as $P_u = \{id_p : p is u’s witness user\}$. For example, in Figure 2(b), the initiator executes three rounds of the finding witness protocol and finds three witness users $\{id_1, id_2, id_3\}$. The red solid lines represent three random walks towards the witness users $id_1$, $id_2$ and $id_3$ respectively. The grey lines represent the pairwise relationships between two users.

B. Verification Stage

In this stage, a user $v$ (the authenticatee) is asked by another user $u$ (the authenticator) to show the pair $(id_u, pk_u)$ is indeed bounded together in a certificate. They invoke the verification protocol (Figure 3). The authenticator and the authenticatee first collaborate to find a common witness user. The authenticatee then provides a zero-knowledge proof of the common user’s certificate. The authenticator is convinced
if the authenticatee successfully passes the Σ-protocol. We delay our system analysis to the next section.

The authenticator and the authenticatee are denoted as u and v respectively.

1) u sends his witness set $P_u$ to v. v randomly picks a witness user’s identity $i_d_u$ from $P_u \cap P_v$ and sends it back to u. The protocol terminates if $P_u \cap P_v = \emptyset$.

2) v then performs a zero-knowledge proof of the certification $(\sigma_1, \sigma_2, r, \sigma_3)_{y \rightarrow v}$ using Σ-protocol derived as in section II-C. We abuse the notation $\sigma_1, \sigma_2, r, \sigma_3_{y \rightarrow v}$ here to denote the HW-signature signed by p on v’s $(i_d_u, pk_v)$.

3) v succeeds if and only if he passes the verification of Σ-protocol.

![Fig. 3. Verification Protocol](image)

C. System Analysis

We note that it is possible $P_u \cap P_v = \emptyset$ which causes the failure in the verification stage. However, our protocol ensures that $P_u \cap P_v \neq \emptyset$ with high probability. Specifically, we have the following theorem:

**Theorem III.1.** Given any two nodes u and v in a fast mixing graph G performing the verification protocol, the probability of $P_u \cap P_v = \emptyset$ is at most $\exp(-m^2/[V]) \cdot (1 - m/[V])$, where $\exp(a)$ denotes $e^a$ and $m$ is the cardinality of the witness user set.

**Proof:** In the building-up stage of our protocol, all of the witness user sets are sampled from the stationary distribution over the node set V. If the stationary distribution is a uniform distribution, then the probability of failure in the verification stage is $(1 - m/[V])^m \leq \exp(-m^2/[V])$ according to the general birthday paradox. Morselli, et al. proved in [14] that for any given distribution, the uniform distribution is the worst case. In other words, the probability of failure is maximized when the stationary distribution is a uniform distribution. Generally speaking, it is proved in [14] that, given two sets $P_u$ and $P_v$ sampling elements from the same distribution over the node set V, the probability of the event that the two sets do not share any common element is $\exp(-(m^2/[V]) \cdot (1 - m/[V]))$ if $m = |P_u| = |P_v|$.

For instance, in order to let the probability of $P_u \cap P_v \neq \emptyset$ be greater than 2/3 in an OSN with 10,000 users, we only need to sample no less than 64 witness users.

Again, we emphasize that the security of our protocol relies on the secure communication between each two adjacent nodes along the random walk. It is easy to see that an honest user u is able to request a valid certificate of the pair $(i_d_u, pk_u)$ and successfully pass the verification stage. Since the secure channel is an authenticated and confidential channel, an adversary $A$ with a public key $pk_A$ is unable to request a certificate on the pair $(i_d_u, pk_A)$ where $i_d_u \neq i_d_A$. According to our building-up protocol, if $A$ wants to initiate a request, he has to set up such a channel with one of his neighbours. However, $A$ is unable to do this on behalf of someone else other than himself because he will fail in the authentication phase. The adversary $A$ is also unable to modify a request or reply messages because all the communication channels along the walk are confidential. Because of the unforgeable property of the signature algorithm, the adversary is unable to fake a certificate.

In the verification stage, the authenticatee does not need to directly show the certificate. He convinces the authenticator by providing the zero-knowledge proof of the certificate. Hence, the adversary is unable to obtain a user’s certificate in the verification stage.

However, a powerful adversary might participate in a social network, establish connections with the honest users and reply the requests from the honest users for signing certificates. Then, the adversary signs a fake pair of $(i_d_u, pk_A)$ using his own signing key and initiates the verification protocol with any user v who ever requested a certificate from $A$ before. Because the adversary $A$ and v can find a common friend $A$ from their witness user sets, they can pass the verification stage. Therefore, $A$ can successfully cheat v who believes that the public key $pk_A$ belongs to u. Now we analyse the number of users $A$ can cheat. A $w$-step random walk will end at $A$ with the probability $P_A$, which is the stationary distribution at the node A. Given a undirected connected graph, $P_A = d_A/(2|E|)$, where $d_A$ is the node $A$’s degree. In a total $m|V|$ requests(every node sends m requests), $A$ is expected to be selected $m \cdot d_A/|V|/|E|$ times, which is the largest number of nodes an adversary could successfully attack. This number is $o(m \cdot d_A)$ in a dense OSN with much more edges than nodes, i.e., $|V| = o(|E|)$, and $d_A$ is usually very small since it is resource costly(e.g. taking a long time) to establish an authenticated link with one node. In practice, this attack exposes the adversary’s identity because the adversary has to establish authenticated communication channels with the honest users in order to participate in the social network. Thus, we argue that the adversaries are not willing to start such an attack because of the social pressure.

Note that our protocol is decentralized and asynchronous. The users do not need to know the topology of the whole OSN but only the local information. It is not necessary to require all the users staying online. The system could store the messages and let the users process them when they are online.

IV. SIMULATION

The performance of our system was analysed in Section III-C. Since the Σ-protocol is the only computational component in the verification protocol, the computational overhead of the verification stage could be theoretically bounded by analysing derived Σ-protocol[2]. Therefore, in this section we mainly evaluated the performance of our finding witness protocol in Kleinberg’s social network model[12]. This synthetic network was used to explain the small world phenomenon (i.e., a person can reach any other person in the world in a few hops) in the social network. Kleinberg’s social network model contains the nodes located in a two-dimensional grid. Given two parameters p and q, every node u in the social network has p local friends and q long distance friends. Local friends are p closest nodes in terms of grid distance. Then u picks up q long distance friends with acceptance probability of $dist(u,v)^{-\tau}$ for each node v, where $\tau$ is a parameter and $dist$ is the grid distance function. In the experiment we constructed a social network with 10,000 nodes and $\tau = 1.8$ (this parameter was also used in [20] and [21]).

Here, we introduce the notion of non-empty intersection. It denotes the intersection of two users’ witness user sets. A high probability of non-empty intersection between any two nodes
is critical to our system. This probability actually depends on two parameters of our system, the length of the random walk \( w \), and the size of the witness user set \( m \). We tested this probability under different parameters \( w \) (from 1 to 16) and \( m \) (from 60 to 180 with 5 increments). The parameters \( p \) and \( q \) were fixed as 10 and 14 respectively. To get the probability, we randomly chose 1,224 pairs of nodes and countered the number of pairs with non-empty intersection. The result was shown in Figure 4. We set \( m = 150 \) when we tested \( w \) and \( m = 120 \) when testing \( m \). First, we note that when \( w > 4 \) or \( m > 130 \), the probability is greater than 90%. We could see that the probability depends largely on the size of our sampling set when \( w > 4 \). It is because the more nodes one user samples, the more probably two users happen to have a common node. However, when \( w < 4 \), the probability decreases sharply until \( w \) goes to 0, because the distribution of the witness users is far away from the stationary distribution. Among those who have non-empty intersections, many users actually share more than one common nodes. Again for those 1,224 pairs who performed a 15-hop random walk and sampled 140 witness users, an inspection on the size of the intersection is provided in Figure 4(c) in the form of the bar chart, in which the bottom axis represents the number of common friends shared by each pair and the height of the bar denotes the number of pairs who have \( x \) common friends.

Fig. 4. Evaluation results with fixed \( p \) and \( q \)

\[ w = 12 \] when testing \( m \). We can see that our system performs better (i.e., higher probability of non-empty intersection) in the network rich in long distance relationships than in local relationships.

V. CONCLUSION

In this paper, to our best knowledge, we for the first time have formalized the problem of authenticating strangers in fast mixing social networks. We have proposed a system for authenticating two strangers in this paper. Based on the small world model, we have proved that two strangers can find a common witness user with a high probability.

ACKNOWLEDGEMENT

This research was supported in part by NSF grant 0901451 and ARO grant W911NF-09-1-0467. The information reported here does not reflect the position or the policy of the federal government.

REFERENCES