POSTER: Privacy-Preserving Profile Similarity Computation in Online Social Networks

Arjan Jeckmans and Qiang Tang and Pieter Hartel
DIES, Faculty of EEMCS
University of Twente, the Netherlands
{a.j.p.jeckmans,q.tang,pieter.hartel}@utwente.nl

ABSTRACT

Currently, none of the existing online social networks (OSNs) enables its users to make new friends without revealing their private information. This leaves the users in a vulnerable position when searching for new friends. We propose a solution which enables a user to compute her profile similarity with another user in a privacy-preserving way. Our solution is designed for a realistic OSN environment, where a pair of users is unlikely to be online at the same time.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems— *Distributed applications*; E.3 [**Data Encryption**]: Public key cryptosystems

General Terms

Algorithms, Human Factors, Security

Keywords

Online social network, matching, privacy

1. INTRODUCTION

Online social networks (OSNs) provide an important platform for users to make new friends and share their information. In reality, OSN users do not fully trust the OSN and users other than their friends, and do not like to disclose their private information to these parties. On the other hand, some OSNs may not want the legal responsibility for storing, processing, and distributing the users' private data.

Consider a Facebook user, Alice, who likes the fan page of artist X. On that page, she finds that Bob also likes artist X. Alice thinks Bob might make a nice friend. To learn more about Bob, she visits his profile. However, Bob's profile is private and Alice learns nothing more about Bob. At this point, Alice has three options:

1. Give up learning more about Bob;
2. Invite Bob to become friends and hope that Bob will make a good friend;
3. Send Bob a private message and wait.

Copyright is held by the author/owner(s).
CCS'11, October 17–21, 2011, Chicago, Illinois, USA.
ACM 978-1-4503-0948-6/11/10.

If Alice follows the first option, she may miss a chance to make a new friend. If Alice follows the second option, her private information may be leaked if she finds out that Bob is not a suitable friend. Another problem with this option is that it is difficult or even impossible to revoke friendship in OSNs. If Alice follows the third option, she may encounter a deadlock when Bob is also cautious about his privacy. In this case, both Alice and Bob do not want to disclose their information to a stranger.

In this paper, we outline a solution, which allows two users to test their profile similarity in OSNs. The solution leverages the chain of friends that exists between two users to preserve their privacy. Our solution captures the following realistic characteristics of existing OSNs:

- A friendship between two users is more likely when their profile similarity is above a certain threshold, where the profile similarity is defined to be the number of common attributes.
- Due to the web-based nature, a (randomly chosen) pair of users is unlikely to be online at the same time [1].
- Users tend to make friends with other users that are connected to them through a chain of friends.

The rest of the poster is organized as follows. In Section 2, we describe our solution and the achieved properties. In Section 3, we describe the related work, and conclude the poster in Section 4.

2. DESCRIPTION OF THE SOLUTION

Ideally, in order to protect users' privacy, we would expect a fully trusted third party (TTP) which can access users' attributes and compute the profile similarity for them. However, such a TTP does not exist in reality and it is always preferable to avoid employing such an entity if possible. For an OSN, the current situation is that it has full access to its users' profile attributes, but we believe that this is undesirable for the users although they do not have a good choice today (they can provide fake attributes though). To solve this problem, our idea is to realize the functionality of a TTP into using two sets of semi-trusted parties: one is the OSN (or an application in OSN), and the other is a chain of friends. As a result, the OSN cannot access users' attributes anymore.

Our solution consists of two phases: a *set up phase* where every user generates its own parameters, and a *profile similarity computation phase* where an online user can compute

her similarity with another online/offline user. The solution works when there is a chain of friendship links between these two users, and we generally assume the friendship is unilateral. If X regards Y as a friend, then it means X allows Y to see the plaintexts encrypted under his public key. Note that, if desired, it can be trivially extended to a bilateral setting.

Due to the space limit, we describe the solution for a simple setting, where there are four users. Suppose that Alice wants to compute her profile similarity with Bob who can be offline. We assume that Bob regards Charlie as a friend, Charlie regards Dave as a friend, and Dave regards Alice as a friend. The solution is briefly summarized in **Figure 1** and detailed below.

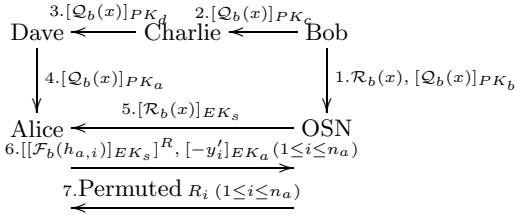

Figure 1: Solution Overview

An extension of this solution to more complex setting is straightforward, we skip the details in this poster.

2.1 Set Up Phase

In this phase, the users and the OSN agree on a group $\mathbb{G}$ of prime order p and a generator g for ElGamal scheme [2]. Let $\mathsf{H}_0 : \{0,1\}^* \to \mathbb{Z}_p$ be a cryptographic hash function.

Alice, Bob, Charlie, Dave generate their ElGamal public/private key pairs $(PK_a = g^{x_a}, SK_a = x_a)$, $(PK_b = g^{x_b}, SK_b = x_b)$, $(PK_c = g^{x_c}, SK_c = x_c)$, and $(PK_d = g^{x_d}, SK_d = x_d)$ respectively, where x_a, x_b, x_c, x_d are randomly chosen from $\mathbb{Z}_p$. Bob, Charlie and Dave compute their proxy re-encryption keys $RK_{b\to c}$, $RK_{c\to d}$, and $RK_{d\to a}$ respectively, where

$$RK_{b\to c} = (\mathsf{H}_0(u_b) - x_b, g^{v_b}, g^{v_b\cdot x_c}\cdot u_b),$$

$$RK_{c\to d} = (\mathsf{H}_0(u_c) - x_c, g^{v_c}, g^{v_c\cdot x_d}\cdot u_c),$$

$$RK_{d\to a} = (\mathsf{H}_0(u_d) - x_d, g^{v_d}, g^{v_d\cdot x_a}\cdot u_d),$$

and u_b, u_c, u_d are randomly chosen from $\mathbb{G}$ and v_b, v_c, v_d are randomly chosen from $\mathbb{Z}_p$.

Note the fact that, instead of directly comparing their attributes, users can firstly pre-process their attributes by hashing them using a cryptographic hash function. Let the hash values fall into a finite field $\mathbb{F}$. Let $\mathsf{H}_1 : \{0,1\}^* \to \mathbb{F}$ be a cryptographic hash function. Suppose that Alice has n_a attributes and the hashed values are $h_{a,i}$ $(1 \le i \le n_a)$, and Bob has n_b attributes and the hashed values are $h_{b,i}$ $(1 \le i \le n_b)$. Bob generates polynomials $\mathcal{F}_b(x)$, $\mathcal{Q}_b(x)$, and $\mathcal{R}_b(x)$ as follows:

$$\mathcal{F}_b(x) = \prod_{i=1}^{n_b}(x - h_{b,i}),\ \mathcal{F}_b(x) = \mathcal{Q}_b(x) + \mathcal{R}_b(x),$$

where the coefficients of $\mathcal{R}_b(x)$ are randomly chosen from $\mathbb{F}$. Let the coefficients of $\mathcal{Q}_b(x)$ be q_j $(0 \le j \le n_b)$, then we denote the encryption of $\mathcal{Q}_b(x)$ as $[\mathcal{Q}_b(x)]_{PK_b}$, which is defined to be

$$(g^{r_j}, g^{r_j\cdot x_b}\cdot t_j, \mathsf{H}_1(t_j) + q_j)$$

for $0 \le j \le n_b$ where r_j, t_j are randomly chosen from $\mathbb{Z}_q$.

The users store the following values at the OSN: PK_a, PK_b, PK_c, PK_d, $RK_{b\to c}$ and $RK_{c\to d}$, $RK_{d\to a}$, $\mathcal{R}_b(x)$, and $[\mathcal{Q}_b(x)]_{PK_b}$. They keep their private keys outside of the OSN.

2.2 Profile Similarity Computation Phase

If Alice wants to compute her profile similarity with Bob, the protocol proceeds in two stages.

Stage 1) Distribution of $\mathcal{Q}_b(x)$ from Bob to Alice

For every $0 \le j \le n_b$, the OSN computes $[\mathcal{Q}_b(x)]_{PK_a}$ by following the following three steps (steps 2-4 in Figure 1):

1. Re-encrypt $[\mathcal{Q}_b(x)]_{PK_b}$ using $RK_{b\to c}$ to get $[\mathcal{Q}_b(x)]_{PK_c}$, which is
$$(g^{r_j}, g^{r_j\cdot \mathsf{H}_0(u_b)}\cdot t_j, \mathsf{H}_1(t_j) + q_j; g^{v_b}, g^{v_b\cdot x_c}\cdot u_b).$$
2. Re-encrypt $[\mathcal{Q}_b(x)]_{PK_c}$ to get $[\mathcal{Q}_b(x)]_{PK_d}$, which is
$$(g^{r_j}, g^{r_j\cdot \mathsf{H}_0(u_b)}\cdot t_j, \mathsf{H}_1(t_j) + q_j; g^{v_b}, g^{v_b\cdot \mathsf{H}_0(u_c)}\cdot u_b;$$
$$g^{v_c}, g^{v_c\cdot x_d}\cdot u_c).$$
3. Re-encrypt $[\mathcal{Q}_b(x)]_{PK_d}$ to get $[\mathcal{Q}_b(x)]_{PK_a}$, which is
$$(g^{r_{j1}}, g^{r_j\cdot \mathsf{H}_0(u_b)}\cdot t_j, \mathsf{H}_1(t_j) + q_j; g^{v_b}, g^{v_b\cdot \mathsf{H}_0(u_c)}\cdot u_b;$$
$$g^{v_c}, g^{v_c\cdot \mathsf{H}_0(u_d)}\cdot u_c; g^{v_d}, g^{v_d\cdot x_a}\cdot u_d).$$

The OSN sends $[\mathcal{Q}_b(x)]_{PK_a}$ to Alice, who can recover $\mathcal{Q}_b(x)$ using her private key x_a.

Stage 2) Two-Party Profile Similarity Computation

In this phase, Alice and the OSN (representing Bob) run a two-party private set intersection protocol, which is an adapted version of that proposed by Freedman et al. [3], to compute the profile similarity. Note that, Alice possesses $\mathcal{Q}_b(x)$ and the OSN possesses $\mathcal{R}_b(x)$ of Bob.

The protocol runs as follows (steps 5-7 in Figure 1).

1. The OSN creates a new ephemeral key pair for the Paillier cryptosystem [6]. Let the public/private key pair be (EK_s, DK_s) and the message space be $\mathbb{Z}_s$. Then, the OSN encrypts $\mathcal{R}_b(x)$ using EK_s to obtain $[\mathcal{R}_b(x)]_{EK_s}$. Note that, the encryption means encrypting every coefficients of $\mathcal{R}_b(x)$. Finally, the OSN sends $[\mathcal{R}_b(x)]_{EK_s}$ to Alice.
2. Recall that Alice's hashed attributes are $h_{a,i}$ $(1 \le i \le n_a)$. With $[\mathcal{R}_b(x)]_{EK_s}$, Alice computes $[\mathcal{R}_b(h_{a,i})]_{EK_s}$ based on the homomorphic property of Paillier cryptosystem (details can be found in [3]). Since Alice possesses $\mathcal{Q}_b(x)$, she can compute $\mathcal{Q}_b(h_{a,i})$ for $1 \le i \le n_a$. Based on the homomorphic property of Paillier cryptosystem, for $1 \le i \le n_a$, Alice can compute $[\mathcal{F}_b(h_{a,i})]_{EK_s}$, where
$$[\mathcal{F}_b(h_{a,i})]_{EK_s} = [\mathcal{R}_b(h_{a,i})]_{EK_s}\cdot[\mathcal{Q}_b(h_{a,i})]_{EK_s}.$$

Then, Alice creates a new ephemeral key pair for the Paillier cryptosystem [6]. Let the public/private key pair be (EK_a, DK_a) and the message space be $\mathbb{Z}_a$. For $1 \leq i \leq n_a$, Alice generates $[[\mathcal{F}_b(h_{a,i})]_{EK_s}]^R$, which is a randomized version of $[\mathcal{F}_b(h_{a,i})]_{EK_s}$, as follows:

$$\begin{aligned}[[\mathcal{F}_b(h_{a,i})]_{EK_s}]^R &= ([\mathcal{F}_b(h_{a,i})]_{EK_s})^{y_i} \cdot [y'_i]_{EK_s} \\ &= [\mathcal{F}_b(h_{a,i}) \cdot y_i + y'_i]_{EK_s}\end{aligned}$$

where y_i and y'_i are randomly chosen values.

Finally, Alice sends $[[\mathcal{F}_b(h_{a,i})]_{EK_s}]^R$ and $[-y'_i]_{EK_a}$ for $1 \leq i \leq n_a$ to the OSN.

3. For $1 \leq i \leq n_a$, the OSN decrypts $[[\mathcal{F}_b(h_{a,i})]_{EK_s}]^R$ to obtain $\mathcal{F}_b(h_{a,i}) \cdot y_i + y'_i$, and computes R_i, where y''_i is randomly chosen and

$$\begin{aligned}R_i &= ([\mathcal{F}_b(h_{a,i}) \cdot y_i + y'_i]_{EK_a} \cdot [-y'_i]_{EK_a})^{y''_i} \\ &= [\mathcal{F}_b(h_{a,i}) \cdot y_i \cdot y''_i]_{EK_a}\end{aligned}$$

Then, the OSN sends a permuted version of R_i ($1 \leq i \leq n_a$).

4. With the permuted version of R_i ($1 \leq i \leq n_a$), Alice decrypts all of them and count the number of "0". Based on this number, which represents the size of joint set of attributes, Alice makes a decision whether she wants to add Bob as a friend or not.

2.3 Properties of the Solution

Assume that every user communicates with the OSN through a secure channel such as SSL/TLS. Based on this assumption, our solution achieves the following desirable properties.

- It supports offline users (say Bob in the solution) by using the OSN as a proxy. This has the additional benefit of preventing (popular) users from being spammed with computation requests.
- The profile privacy of Alice is guaranteed by the protocol. In the solution, Alice reveals $[[\mathcal{F}_b(h_{a,i})]_{EK_s}]^R$ and $[-y'_i]_{EK_a}$ for $1 \leq i \leq n_a$. Suppose that the OSN is a semi-honest, it learns nothing about Alice's attributes $h_{a,i}$ ($1 \leq i \leq n_a$). Certainly, Bob and any other users learn no information.
- The profile privacy of Bob relies on splitting the polynomial $\mathcal{F}_b(x)$ into $\mathcal{Q}_b(x)$ and $\mathcal{R}_b(x)$ and grant them to Alice and the OSN separately. The OSN possesses $\mathcal{R}_b(x)$ but does not have access to $\mathcal{Q}_b(x)$ which is protected by the re-encryption scheme described in Stage 1 in the profile similarity computation. Alice possesses $\mathcal{Q}_b(x)$ but has no access to $\mathcal{R}_b(x)$. Note that neither $\mathcal{Q}_b(x)$ nor $\mathcal{R}_b(x)$ alone will reveal any information about Bob's hashed attributes. Furthermore, based on the security of the adapted protocol from [3], if the OSN is semi-honest, we can conclude that neither the OSN nor any other user will learn anything about Bob's hashed attributes.

3. RELATED WORK

Tang [7] proposed a privacy-preserving profile matching protocol based on fuzzy extractor and CAPTCHA. The drawback of this approach is that a semi-honest can recover information about users' attributes if it can solve the employed CAPTCHA scheme. There are works which attempt to protect user information in OSNs, although they do not directly tackle with privacy-preserving profile matching solutions. Some of them are the following. Lucas and Borisov [5] proposed to use public key encryption to send messages between users based on proxy re-encryption techniques. Guha et al. [4] proposed to mix user information across profiles to minimize the link-ability of information, while leaving users with no way to verify profile correctness, unless some information is already known. Tootoonchian et al. [8] proposed to store profile information at a trusted place and access is then controlled using relationship certificates.

4. CONCLUSION

We propose a solution for a pair of OSN users to compute their profile similarity. Our solution gives users a new option in searching for new friends without losing their privacy over their private data. A detailed discussion of our solution will be given in the full paper. If the OSN is not semi-honest, then it may try to collude with Bob's friends. In this case, the OSN will be able to recover Bob's hashed attributes, based on which a brute-force attack will reveal the attributes. How to deter a malicious OSN is an interesting future work. In addition, if many users initiate the protocol, the OSN needs to perform a lot of ElGamal re-encryption and Paillier encryption/decryption. This may become a burden, and how to improve the efficiency is another future research topic.

5. REFERENCES

[1] F. Benevenuto, T. Rodrigues, M. Cha, and V. A. F. Almeida. Characterizing user behavior in online social networks. In *Internet Measurement Conference*, pages 49–62, 2009.

[2] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In G. R. Blakley and D. Chaum, editors, *CRYPTO '84*, volume 196 of *LNCS*, pages 10–18, 1985.

[3] M. J. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In *EUROCRYPT '04*, pages 1–19, 2004.

[4] S. Guha, K. Tang, and P. Francis. Noyb: privacy in online social networks. In *Proceedings of the first workshop on Online Social Networks*, pages 49–54, 2008.

[5] M. M. Lucas and N. Borisov. Flybynight: mitigating the privacy risks of social networking. In *Proceedings of the 7th ACM workshop on Privacy in the electronic society*, pages 1–8, 2008.

[6] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT '99*, pages 223–238, 1999.

[7] Q. Tang. User-friendly matching protocol for online social networks. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 732–734, 2010.

[8] A. Tootoonchian, S. Saroiu, Y. Ganjali, and A. Wolman. Lockr: better privacy for social networks. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 169–180, 2009.