User-Friendly Matching Protocol for Online Social Networks

Qiang Tang
DIES, Faculty of EEMCS, University of Twente
Enschede, the Netherlands
q.tang@utwente.nl

ABSTRACT

In this paper, we outline a privacy-preserving matching protocol for OSN (online social network) users to find their potential friends. With the proposed protocol, a logged-in user can match her profile with that of an off-line stranger, while both profiles are maximally protected. Our solution successfully eliminates the requirement of "out-of-band" communication channels, which is one of the biggest obstacles facing cryptographic solutions for OSNs.

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

OSNs (online social networks) have gained a great popularity in recent years, and become an important communication platform for Netizens. Popular OSNs, such as Facebook and Myspace, have attracted millions of users. The core attraction of OSNs is that they enable a user to: (1) construct a profile to represent herself within the OSN; (2) build up connections with other users through matching their profiles and share their information afterwards. In addition, OSNs often provide some additional applications, developed by either OSNs themselves or third parties, to encourage users' engagement in their networks.

Generally, the system structure of an OSN is depicted in Figure 1, where the dashed line means friendship connection. For a user, say Alice, all her information (profile, friendship graph, etc) is stored in plaintext by the OSN. When Alice logs into her account, she can surf the whole network and look into another user's profile. If Alice decides that Bob's profile is interesting, then she can then add Bob as her friend.

CCS'10, October 4–8, 2010, Chicago, Illinois, USA.
ACM 978-1-4503-0244-9/10/10.

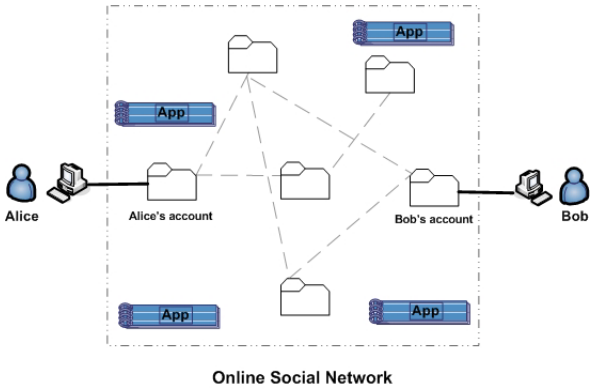

Figure 1: OSN system Structure

1.1 Motivation

Numerous academic analysis and reality incidents have shown that users' profile information can be easily disclosed to the unintended audience. In practice, unintended information disclosure can result in very negative consequences, e.g. identity theft, blackmailing, and stalking. To cope with the ever-deteriorating situation, most OSNs have provided some privacy settings. Take Facebook as an example, for each of her profile items, a user can restrict it to be accessible only by Friends, Friends of Friends, or Everyone.

While the implemented privacy settings help users gain better control over their information, they can interfere with existing OSN functionalities. Let's consider the following example situation occurred in Facebook. When Alice uses an application X, she notices that Bob has also used the same application recently. To learn more information about Bob, Alice visits Bob's page and finds nothing except the advice given by Facebook: *"Bob only shares some of his profile information with everyone. If you know Bob, send him a message or add him as a friend."*. In this situation, Alice has three choices.

1. Alice gives up without trying to learn anything more about Bob. In this case, Alice could have missed a chance to meet Bob who has similar interests to hers, so is for Bob.

2. Alice adds Bob as her friend. In this case, later on, Alice may figure out that she does not have many common interests with Bob and does not want to share her

information with Bob. Alice may try to remove Bob from her friend list, but it is an awkward task.

3. Alice writes Bob a message to introduce herself and ask for more information. In this case, if both Alice and Bob want to protect their profile information, they need a secure mechanism (not existing) to fairly reveal information to each other.

The illustrated interference indeed reflects the tension between the OSN functionalities and users' privacy expectations in the scope of an OSN. On one hand, users want to acquire as much information as possible from others in order to find friends. On the other hand, scared by the potential privacy breaches, users tend to hide as much information as possible from others.

One more related concern is that, regardless of the privacy settings, users' profile information is always in plaintext for the OSN. How to prevent a malicious OSN from abusing users' information is still an unsolved issue.

1.2 Contribution

With respect to protecting users' profile information inside an OSN, we adopt the following threat model. A user shares her profile information with her friends only, i.e. any unauthorized entity is considered as a potential adversary. The OSN is semi-trusted in the following sense: it will faithfully mediate the protocol execution but is likely to mount an *automated information recovery* attack, through which the OSN employs computers to recover users' information without massive involvement of humans. The motivation behind this is that, for an OSN with millions of users, the main concern is that the OSN may massively collect and mine its users' profile information, while it is less concerned that the OSN will mount a targeted attack against a specific user.

We tackle the identified problem by introducing a novel privacy-preserving protocol for OSN users to match their profiles. Our proposal has the following unique features.

1. It assumes a general environment, where Alice logs into her account and tries to match her profile with that of Bob, who is off-line at the moment. Certainly, the protocol also works when Bob has logged in.

2. Users' profile information is encrypted by self-generated private keys, and no TTP is required to perform key certification. CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) [1] is employed to "hide" cryptographic keys from the OSN and verify a human user's presence.

3. In contrast to other schemes, e.g. [2, 3], by employing a fuzzy extractor [4], our proposal does not require any "out-of-band" communication channel for users to exchange private keys.

2. THE PROPOSED SOLUTION

In our proposal, users' profile information will be encrypted and the matching will be carried out based on the encrypted data. Therefore, we will use PKE (public key encryption) schemes with additively homomorphic property, e.g. Paillier [5]. It is worth noting that a fully-homomorphic encryption scheme will be a better option, however, existing solutions are not practical yet. The other cryptographic building block is fuzzy extractor [4], defined as follows.

DEFINITION 1. *A $(U, m, \ell, t, \epsilon)$ fuzzy extractor consists of two polynomial-time algorithms* $(\mathsf{Gen}, \mathsf{Rep})$.

- Gen*:* $U \rightarrow R \times \{0,1\}^*$. *This algorithm takes* $u \in U$ *as input, and returns a secret* $r \in R = \{0,1\}^\ell$ *and a public helper data* $p \in \{0,1\}^*$.
- Rep*:* $U \times \{0,1\}^* \rightarrow R$. *This algorithm takes* $u' \in U$ *and* $p \in \{0,1\}^*$ *as input, and returns a string from* R.

Informally, the Gen algorithm extracts an ephemeral secret r from a long-term secret u, while the Rep algorithm can recover the ephemeral secret r from the public helper data p given u', whose distance to u is within the threshold t.

Besides these cryptographic primitives, a CAPTCHA scheme will be employed. An instance of an example CAPTCHA scheme is shown in Figure 2, and it is in fact an image embedded with the English words "following finding". The CAPTCHA scheme is secure if a computer cannot recognize the words in the image with a high probability. CAPTCHA schemes have been widely used to deter automated attacks in a number of contexts, such as web account registration and account login.

Figure 2: Example of CAPTCHA

It is worth noting that CAPTCHA schemes don't provide security guarantees as strong as those by cryptographic primitives. Some CAPTCHA schemes have been broken, e.g. [6]. Nonetheless, they are still a popular security measure.

2.1 Description of the Solution

We assume that the OSN provides an application MApp for profile matching. The MApp selects a symmetric key encryption scheme $(\mathsf{SEnc}, \mathsf{SDec})$ and a cryptographic hash function H. As in [7], for a user, the running applications in her local computer are assumed to be trustworthy.

Without loss of generality, suppose that Alice and Bob have n profile items, denoted by a_i $(1 \leq i \leq n)$ and b_i $(1 \leq i \leq n)$ respectively. Alice generates a public/private key pair (pk_a, sk_a) for an additively homomorphic PKE scheme $(\mathsf{PEnc}, \mathsf{PDec})$, and generates a CAPTCHA instance cap_a by embedding her private key sk_a in an image. Instead of her plaintext profile, Alice stores the following information under her account in the OSN:

$$p_a,\ \mathsf{H}(r_a),\ pk_a,\ cap_a,\ \mathsf{SEnc}(pk_a||sk_a, r_a),\ \mathsf{PEnc}(a_i, pk_a)\ (1 \leq i \leq n),$$

where $(r_a, p_a) = \mathsf{Gen}(\{a_1, a_2, \cdots, a_n\})$, and $(\mathsf{Gen}, \mathsf{Rep})$ is a fuzzy extractor scheme with threshold *set difference distance* value t_a. Similarly, Bob generates a public/private key pair (pk_b, sk_b) and generates a CAPTCHA instance cap_b by embedding his private key sk_b in an image. Bob stores the following information under his account in the OSN:

$$p_b,\ \mathsf{H}(r_b),\ pk_b,\ cap_b,\ \mathsf{SEnc}(pk_b||sk_b, r_b),\ \mathsf{PEnc}(b_i, pk_b)\ (1 \leq i \leq n),$$

where $(r_b, p_b) = \mathsf{Gen}'(\{b_1, b_2, \cdots, b_n\})$, and $(\mathsf{Gen}', \mathsf{Rep}')$ is a fuzzy extractor scheme with threshold *set difference distance*

value t_b. We stress that t_a (t_b) serves as a threshold value for being matched by another user when Alice (Bob) is off-line.

Motivated by the fact that friendship is something of reciprocal interest to both involved parties, we consider matching to be a two-stage process. Suppose Alice logs into her account and wants to match her profile with that of Bob. The first stage (steps 2–4 of the proposed protocol) is to determine whether Alice is eligible to become a friend of Bob according to Bob's criteria, which requires that the *set difference distance* between $\{a_1, a_2, \cdots, a_n\}$ and $\{b_1, b_2, \cdots, b_n\}$ is at most t_b. The second stage (steps 5–7 of the proposed protocol) is to determine whether Bob is qualified to be Alice's friend according to Alice's criteria, which is defined to be the *set difference distance* between $\{a_1, a_2, \cdots, a_n\}$ and $\{b_1, b_2, \cdots, b_n\}$ is at most t'_a. It is worth noting that t'_a may not be equal to t_a since they represent Alice's threshold values for the online and off-line situations respectively.

In more details, the protocol works as follows.

1. Alice logs into her account and sends a request to the MApp for a matching with Bob. Meanwhile, Alice downloads her information to her local computer. In particular, Alice recovers her private key sk_a from cap_a and profile items a_i $(1 \leq i \leq n)$ from $\mathsf{PEnc}(a_i, pk_a)$ $(1 \leq i \leq n)$.
2. The MApp fetches p_b and $\mathsf{H}(r_b)$ from Bob's account and sends p_b to Alice.
3. Alice sends $\mathsf{H}(r'_b)$ to the MApp, where
$$r'_b = \mathsf{Rep}'(\{a_1, a_2, \cdots, a_n\}, p_b).$$
4. The MApp terminates the protocol if $\mathsf{H}(r'_b) \neq \mathsf{H}(r_b)$. Otherwise, it fetches $\mathsf{SEnc}(pk_b||sk_b, r_b)$ from Bob's account, generates a CAPTCHA instance cap_t by embedding the fetched information in an image, and sends cap_t to Alice.
5. Alice first recovers $\mathsf{SEnc}(pk_b||sk_b, r_b)$ from cap_t and obtains (pk_b, sk_b) by decrypting it using r'_b. Alice sends $\mathsf{PEnc}(a_i, pk_b)$ $(1 \leq i \leq n)$ to the MApp.
6. The MApp fetches $\mathsf{PEnc}(b_i, pk_b)$ $(1 \leq i \leq n)$ from Bob's account, computes $\mathsf{PEnc}(a_i - b_i, pk_b)$ $(1 \leq i \leq n)$, and sends a re-randomized and permuted version of these values to Alice. Let these values be denoted as R_i $(1 \leq i \leq n)$.
7. Alice decrypts R_i $(1 \leq i \leq n)$ and counts the total number x of 0s. If $n - x \leq t'_a$, she adds Bob as a friend.

From the description, it is clear that Bob is not required to get involved in person, while the matching only relies on his stored information under his account in the OSN.

2.2 Security Analysis

Here, we briefly analyze the security of the proposed protocol with respect to three categories of adversaries.

The MApp has access to the obscured private keys of Alice or Bob, namely cap_a and cap_b, but it cannot automatically recover sk_a and sk_b if the employed CAPTCHA scheme is secure. If a user wants stronger security, then she can encrypt her private key first using a password and then generate a CAPTCHA instance by embedding the ciphertext in an image.

With the above protocol, Alice learns Bob's private key sk_b if Alice is allowed to add Bob as a friend according to Bob's criteria, namely the set difference distance between $\{a_1, a_2, \cdots, a_n\}$ and $\{b_1, b_2, \cdots, b_n\}$ is at most t_b. Therefore, this is nothing more than that Bob shares his profile information with his potential friend. In step 4 of the protocol, the information $\mathsf{SEnc}(pk_b||sk_b, r_b)$ is sent to Alice via a CAPTCHA instance, so that it verifies Alice's presence and prevents a computer program from mounting automated attacks. Therefore, it is secure for Bob.

Clearly, in the protocol execution, Bob learns nothing about Alice. Therefore, it is secure for Alice.

3. CONCLUSION

We have briefly outlined a matching protocol to match potential friends in OSNs. By using the techniques such as fuzzy extractor and CAPTCHA, the protocol is user-friendly and secure against the common attacks. The precise security and performance analysis and potential improvements are an ongoing research work in the Kindred Spirits project.

Acknowledgement

This is an ongoing work carried out in the Kindred Spirits project (http://www.ksproject.nl/), which is sponsored by STW's Sentinels program in the Netherlands. The author would like to thank Michael Beye, Arjan Jeckmans, Sandeep Kumar, and Pieter Hartel for their comments.

4. REFERENCES

[1] L. von Ahn, M. Blum, and J. Langford, "Telling humans and computers apart automatically," *Commun. ACM*, vol. 47, no. 2, pp. 56–60, 2004.

[2] M. J. Freedman and A. Nicolosi, "Efficient private techniques for verifying social proximity," in *Proceedings of the 6th International Workshop on Peer-to-Peer Systems*, 2007.

[3] S. Guha, K. Tang, and P. Francis, "NOYB: privacy in online social networks," in *Proceedings of the first workshop on Online Social Networks*. ACM, 2008, pp. 49–54.

[4] Y. Dodis, L. Reyzin, and A. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," in *Advances in Cryptology - EUROCRYPT 2004*, ser. LNCS, C. Cachin and J. Camenisch, Eds., vol. 3027, 2004, pp. 523–540.

[5] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology, Proceedings of EUROCRYPT '99*, ser. LNCS, J. Stern, Ed., vol. 1592, 1999, pp. 223–238.

[6] G. Mori and J. Malik, "Recognizing objects in adversarial clutter: Breaking a visual CAPTCHA," *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 1, p. 134, 2003.

[7] 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 (WPES)*, 2008, pp. 1–8.