Towards a Privacy-Preserving Solution for OSNs

Qiang Tang

APSIA group, SnT, University of Luxembourg
6, rue Richard Coudenhove-Kalergi, L-1359 Luxembourg
qiang.tang@uni.lu

Abstract. In this short paper, we describe a solution to protect users' privacy in online social networks (OSNs). The solution achieves the following functionalities: (1) it enables users to store their private data securely; (2) it enables users, from the same or different OSNs, to compute their similarity through a secure protocol; (3) it enables similar users to establish a session key for secure communication. Different from existing solutions in the literature, which often rely on a global public key infrastructure or/and traditional key distribution techniques, the proposed solution leverages on the trust between friends and the entropy of users' private attributes.

1 Introduction

Online social networks (OSNs) provide the service that connects classmates, friends, and other people who share similar interests and activities across political, economic, and geographic borders. As surveyed in [1], a large number of OSNs exist, among which Facebook, MySpace, Google +, and Twitter are the most popular ones.

Due to their nature, OSNs can easily collect a huge amount of user data. Among all kinds of OSN data, a particularly important one is profile attributes. In most OSNs, profile attributes consist of a lot of information, ranging from name, address, education background to political views, hobbies, and daily activities. All attributes are available in plaintext to the OSN service providers and, depending on the configurations, some of them are available to third parties. It is not surprising that a subset of the profile attributes can already identify a user, even after anonymization [4, 6]. Therefore, it is an interesting task to design a solution for users to: (1) protect their private profile attributes; (2) establish friendship with strangers based on their profile similarities (this is the main reason why users want to publish their profiles). This implies that the solution should partially resolve the privacy-functionality tension [8], by simultaneously providing privacy protection for profile attributes and allowing users to conveniently compute their profile similarities.

In reality, it is reasonable to assume that most users are involved in multiple OSNs. Now, suppose that both Alice and Bob are enrolled in Facebook and Myspace, and they have the same location attribute in Facebook and the same music taste attribute in Myspace. Due to the different focuses of the OSNs, Alice and Bob may not disclose their location information in Myspace, at the same time they may not disclose their music taste information in Facebook. It will not be a surprise that Alice and Bob are not friends in Facebook and Myspace, because they do not share much in common in either of the OSNs. If they realize their common attributes in both OSNs, Alice and Bob may like to consider each other as a friend and attend some music event together in the city. This indicates that it is desirable to have a solution which works across multiple OSNs.

1.1 Our Contribution

The contribution of this paper is threefold. Firstly, we describe some new cryptographic building blocks and briefly analyze their security properties, including a unilateral set intersection cardinality protocol and a unilateral comparison protocol. Secondly, we propose a solution for protecting users' private profile attributes in OSNs. In the solution, a transitive and uni-directional proxy re-encryption scheme [5] allows users to encrypt their private profile attributes with their own public keys. Based on the unilateral set intersection cardinality protocol, we design an Online-Offline profile matching protocol, which allows two users to compute their profile similarity and one of them can stay offline. Based on the unilateral comparison protocol and a fuzzy extractor scheme [3], we design an Online-Online profile matching protocol, which allows two online users to compute their profile similarity. Thirdly, we observe that users' communications are under surveillance by the OSN service providers. So, we propose a secure channel establishment protocol which allows two users to exchange a session key if they share a certain number of common private profile attributes.

1.2 Organization

The rest of this paper is organized as follows. In Section 2, we describe the new building blocks which will be used later on. In Section 3, we briefly describe the proposed solution. In Section 4, we provide the details of the profile matching protocols and a secure channel establishment protocol, employed in the proposed solution. In Section 5, we conclude the paper.

2 New Cryptographic Building Blocks

The proposed solution employs transitive and uni-directional proxy re-encryption cryptosystem , namely (KeyGen, Enc, Dec, Pextract, Preenc) [5], and the following two new protocols.

2.1 Unilateral Set Intersection Cardinality Protocol

Let ℓ be the security parameter, $n > 1$ be an integer and $\mathbb{F} = \mathbb{Z}_q$ where q is a prime number (i.e. $\mathbb{F}$ is a finite field). We assume that the bit-length of q is a polynomial of the security parameter ℓ and $n < q$. Consider the following client-server setting: the server possesses a polynomial $\mathcal{R}(x) \in \mathbb{F}[x]$; the client possesses a polynomial $Q(x) \in \mathbb{F}[x]$ and c_i $(1 \leq i \leq n) \in \mathbb{F}$. Suppose that $\mathcal{R}(x)$ and $Q(x)$ are of degree n, and the roots of $\mathcal{F}(x) = \mathcal{R}(x) + Q(x)$ are denoted as $d_i(1 \leq i \leq n) \in \mathbb{F}$. Based on Paillier scheme [7], the following protocol allows the client to learn the cardinality of the set intersection between c_i $(1 \leq i \leq n)$ and d_i $(1 \leq i \leq n)$, while the server learns nothing.

1. The server generates a paillier key pair (PK_s, SK_s), where the public key is $PK_s = (N_s, g_s)$. The client generates a paillier key pair (PK_c, SK_c), where the public key is $PK_c = (N_c, g_c)$. then, they exchange and validate their public keys. Here, we assume that $q^5 < N_s$ and $q < N_c$, so that the polynomial coefficients and roots can be directly encrypted by both public keys.
2. The server encrypts its polynomial $\mathcal{R}(x)$ and sends the ciphertext $[\mathcal{R}(x)]_{PK_s}$ to the client. Note that $[\mathcal{R}(x)]_{PK_s}$ is a vector, consisting of the ciphertexts of $\mathcal{R}(x)$'s coefficients under PK_s.
3. For every attribute c_i $(1 \leq i \leq n)$, the client does the following: (1) compute $[\mathcal{R}(c_i)]_{PK_s}$ based on $[\mathcal{R}(x)]_{PK_s}$ and c_i; (2) compute $Q(c_i)$ and its ciphertext $[Q(c_i)]_{PK_s}$; (3) compute $[\mathcal{F}(c_i)]_{PK_s}$ based on $[\mathcal{R}(c_i)]_{PK_s}$ and $[Q(c_i)]_{PK_s}$; (4) select $y_i \in_R \mathbb{Z}_{q^4}$ and compute the randomized value $[\mathcal{F}(c_i) + y_i]_{PK_s}$; (5) compute $y'_i = y_i \mod q$ and $[N_c - y'_i]_{PK_c}$. After all the computations, the client sends $\mathcal{F}(c_i) + y_i]_{PK_s}$, $[N_c - y'_i]_{PK_c}$ $(1 \leq i \leq n)$ to the server.
4. After receiving the values from the client, for every i $(1 \leq i \leq n)$, the server does the following: (1) decrypt $[\mathcal{F}(c_i) + y_i]_{PK_s}$ to obtain $\mathcal{F}(c_i) + y_i$; (2) compute $T_i = \mathcal{F}(c_i) + y_i \mod q$ which is equal to $\mathcal{F}(c_i) + y'_i \mod q$; (3) select $y''_i \in_R N_c$ and compute $R_i = ([T_i]_{PK_c} \cdot [N_c - y'_i]_{PK_c})^{y''_i} \mod N_c^2$. After all the computations, the server sends a randomly permuted version of $\{R_i \ (1 \leq i \leq n)\}$ to the client.
5. The client decrypts R_i $(1 \leq i \leq n)$, and count the number of 0s as the intersection size.

2.2 Unilateral Comparison Protocol

Let $\mathbb{G}$ be a group of prime order p, and $\mathsf{H}_2 : \{0,1\}^* \to \mathbb{G}$ and $\mathsf{H}_3 : \{0,1\}^* \to \{0,1\}^\ell$ be two hash functions. If a client wants to test whether his value S is equal to the value S' of the server, then the client initiates the protocol shown in Fig. 1.

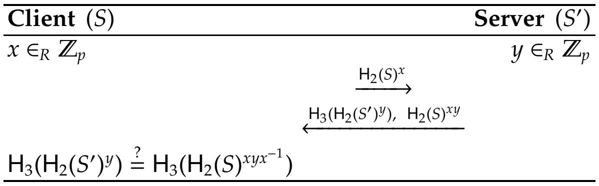
Client (S)
Server (S')
$x \in_R \mathbb{Z}_p$
$y \in_R \mathbb{Z}_p$
$\mathsf{H}_2(S)^x$
$\mathsf{H}_3(\mathsf{H}_2(S')^y),\ \mathsf{H}_2(S)^{xy}$
$\mathsf{H}_3(\mathsf{H}_2(S')^y) \overset{?}{=} \mathsf{H}_3(\mathsf{H}_2(S)^{xyx^{-1}})$

Fig. 1. Unilateral Comparison Protocol

3 The Proposed Solution for OSNs

We generally assume that there is a semi-trust relationship among friends in OSNs. By "semi-trust", we mean that if Alice semi-trusts Bob then she can assume that Bob will not collude with a third-party or reveal her private information. Moreover, we assume that the semi-trust relationship is unilateral, which means that "Alice semi-trusts Bob" does not immediately imply "Bob semi-trusts Alice". Furthermore, we assume that the semi-trust relationship is transitive: if Alice semi-trusts Bob, Bob semi-trusts Charlie, then Alice will semi-trust Charlie.

There is a PPCP server, which is semi-trusted to every user in the system. Therefore, users do not need to fully trust the PPCP server to store their plaintext attributes. Compared with any current OSN, where users need to fully trust the service providers, this is an improvement. Every user can communicate with the PPCP server through a secure channel. Moreover, the PPCP server is trusted to publish the following parameters, used by all users.

- Security parameter: ℓ.
- ElGamal parameter: a multiplicative group $\mathbb{G}$ of degree p, a generator g, and three cryptographic hash functions $\mathsf{H}_0 : \mathbb{G} \rightarrow \mathbb{Z}_p$, $\mathsf{H}_1 : \{0,1\}^* \rightarrow \{0,1\}^L$, $\mathsf{H}_2 : \{0,1\}^* \rightarrow \mathbb{G}$, and $\mathsf{H}_3 : \{0,1\}^* \rightarrow \{0,1\}^\ell$ where L is a polynomial of the security parameter.
- Profile encapsulation parameter: a finite field $\mathbb{F} = \mathbb{Z}_q$ where q is a prime number. We assume that the attributes fall into $\mathbb{F}$.

Let all the users be denoted as U_i $(1 \leq i \leq N)$, where N is an integer, and U_i's attributes be denoted as $\mathcal{A}_i = \{h_{i,j}\ (1 \leq j \leq n)\}$. The proposed solution is composed of three services, including the secure profile storage service, the secure profile matching service, and the secure communication service. They are described in detail below.

3.1 Secure Profile Storage Service

U_i registers at the PPCP server and obtains an identifier ID_i. Moreover, U_i generates an ElGamal public/private key pair (PK_i, SK_i), where $(SK_i = x_i, PK_i = g^{x_i})$, following the specification in [5] based on the ElGamal parameter. U_i sends the public parameters (ID_i, PK_i) to its friends.

1. U_i chooses a subset of his friends that he semi-trusts, denoted as U_{i_x} $(1 \leq x \leq N_i)$.
2. U_i performs the following operations.
   (a) Generate re-encryption keys $RK_{i \rightarrow i_x}$ for every $(1 \leq x \leq N_i)$, which is identified by (ID_i, ID_{i_x}).
   (b) Based on his attributes $h_{i,j}$ $(1 \leq j \leq n)$, generate $\mathcal{F}_i(x), \mathcal{Q}_i(x), \mathcal{R}_i(x) \in \mathbb{F}[x]$ of degree n as follows: $\mathcal{F}_i(x) = \prod_{j=1}^{n}(x - h_{i,j})$, $\mathcal{F}_i(x) = \mathcal{Q}_i(x) + \mathcal{R}_i(x)$, where the coefficients of $\mathcal{R}_i(x)$ are randomly chosen from $\mathbb{F}$.

(c) Based on the ElGamal encryption algorithm Enc specified in [5], encrypt $\mathcal{Q}_i(x)$ using PK_i to obtain $[\mathcal{Q}_i(x)]_{PK_i} = (g^{r_i}, g^{r_i \cdot x_i} \cdot t_i, \mathsf{H}_1(t_i) \oplus \underline{\mathcal{Q}_i(x)})$, where $r_i \in_R \mathbb{Z}_p$, $t_i \in_R \mathbb{G}$, $\underline{\mathcal{Q}_i(x)}$ represents the coefficients of $\mathcal{Q}_i(x)$.

3. U_i stores (PK_i, $RK_{i \to i_x}$ $(1 \leq x \leq N_i)$, $\mathcal{R}_i(x)$, $[\mathcal{Q}_i(x)]_{PK_i}$) at the PPCP server, and associates the data to his identifier ID_i. He keeps SK_i private locally.

With users' data, the PPCP server can construct a social graph $\mathcal{G}$ of the semi-trust relationships among users. In this graph, there is a directed edge from U_i to U_j if U_i semi-trusts U_j (i.e. U_i has generated a re-encryption key $RK_{i \to j}$).

3.2 Secure Profile Matching Service (i.e. Friendship Establishment)

Suppose that U_j has obtained some public information about U_i and consider him as a potential friend. For example, U_i may have publish his identifier ID_i and some hobby information at Facebook, and U_j surfed to U_i's page and obtained the information. Then, U_j can send ID_i to the PPCP server and request to match with U_i. When the PPCP server receives a request, it first checks whether U_i is online. If so, it check U_i's policy, which can have two possibilities.

1. If U_i prefers to run the Online-Online protocol described in Section 4.2 when he is online, then U_j and U_i run the protocol.
2. If U_i prefers not to be involved in the matching, the PPCP server tries to find the shortest semi-trust link from U_i to U_j. If the length of the link is within a threshold agreed by U_i, then the PPCP server represents U_i to run the Online-Offline protocol described in Section 4.1 with U_j. Otherwise, U_j's request is rejected.

If U_i is offline, the PPCP server checks U_i's policy to see whether he wants his profile to be matched when he is offline. If so, the PPCP server does the same as in the aforementioned possibility 2. Otherwise, U_j's request is rejected.

3.3 Secure Communication Service

Suppose that there is a semi-trust link from U_j to U_i, and these two users want to protect their communications. Then, then they can run the secure channel establishment protocol described in Section 4.3. Note that the existence of semi-trust link implies that U_j and U_i share a certain number of common profile attributes, therefore, the protocol will generate a common session key for them.

4 The Employed Protocols

In this section, we describe two profile matching protocols and a secure channel establishment protocol, that are refereed to in the previous section.

4.1 Online-Offline Matching Protocol

Suppose that a user U_j wants to match his profile with U_i and there is a semi-trust link from U_i to U_j, namely there is a chain of proxy re-encryption keys $(RK_{i \to i_1}, RK_{i_1 \to i_2}, \cdots, RK_{i_t \to j})$ from U_i to U_j. In this case, the following protocol is carried out between U_j and the PPCP server.

1. In the first stage, the polynomial $Q_i(x)$ is transferred to U_j. In more detail, the PPCP server performs a series of re-encryptions to transform $[Q_i(x)]_{PK_i}$ into $[Q_i(x)]_{PK_j}$ using the chain of re-encryption keys. From $[Q_i(x)]_{PK_j}$, U_j can recover $Q_i(x)$ using his own private key SK_j. At the end of this stage, U_j has $Q_i(x)$ and his own attributes $\mathcal{A}_j = \{h_{j,t}\ (1 \leq t \leq n)\}$, and the PPCP server possess $\mathcal{R}_i(x)$.
2. In the second stage, U_j and the PPCP server run the unilateral set intersection cardinality protocol, specified in Section 2.1, where U_j and the PPCP server play the roles of the client and the server respectively. At the end of the protocol execution, U_j learns her profile simplicity with U_i.

4.2 Online-Online Matching Protocol

The proposed protocol makes use of a $(\mathcal{U}, \ell_1, \ell_2, t, \epsilon)$-fuzzy extractor [3], where $\mathcal{U}$ is the domain of profile attribute set. When U_j is the initiator, the proposed protocol proceeds in two stages.

1. In the first stage, U_j and U_i engage in a protocol, shown in Fig. 2, where H_1 is defined in Section 3.

$$\begin{array}{lcr}
\hline
\mathbf{U_j} & & \mathbf{U_i} \\
(\mathcal{A}_j = \{h_{j,t}\ (1 \leq t \leq n)\}) & & (\mathcal{A}_i = \{h_{i,t}\ (1 \leq t \leq n)\}) \\
\hline
(r_j, h_j) = \mathsf{Gen}((h_{j,1}, \cdots, h_{j,n})) & & \\
 & \xrightarrow{h_j} & \\
 & & (r_i, h_i) = \mathsf{Gen}((h_{i,1}, \cdots, h_{i,n})) \\
 & \xleftarrow{h_i} & \\
r'_i = \mathsf{Rep}(\mathcal{A}_j, h_i) & & r'_j = \mathsf{Rep}(\mathcal{A}_i, h_j) \\
ck_j = \mathsf{H}_1(ID_i||ID_j||h_i||r'_i||h_j||r_j) & & ck_i = \mathsf{H}_1(ID_i||ID_j||h_i||r_i||h_j||r'_j) \\
\hline
\end{array}$$

Fig. 2. Online Matching Protocol (Stage 1)

2. In the second stage, U_j initiates the unilateral comparison protocol, specified in Section 2.2, to test whether $ck_j = ck_i$.

4.3 Secure Channel Establishment Protocol

As in Section 4.2, the proposed protocol combines a $(\mathcal{U}, \ell_1, \ell_2, t, \epsilon)$-fuzzy extractor scheme [3] and a secure password-based authenticated key exchange

(PAKE) scheme. Note that a lot of PAKE schemes exist in the literature, Boyd and Mathuria [2] provided a survey for those proposed before 2004. In more detail, when U_j initiates the protocol with U_i, then they perform as follows.

1. In the first stage, they run the protocol shown in Fig. 2 to establish some ephemeral secrets. U_j generates $ck_j = \mathsf{H}_1(ID_i||ID_j||h_i||r'_i||h_j||r_j)$ and U_i generates $ck_i = \mathsf{H}_1(ID_i||ID_j||h_i||r_i||h_j||r'_j)$. Note that if the distance between $\mathcal{A}_i$ and $\mathcal{A}_j$ is smaller than t, then $ck_j = ck_i$.
2. In the second stage, they run a secure PAKE scheme to establish a session key. The key materials of ck_j and ck_i are used as the passwords.

5 Conclusion

In this short paper, we have briefly outlined a privacy-preserving solution for OSNs. The solution provides three services, including the secure profile storage service, the secure profile matching service, and the secure communication service. More details about the proposed solution and the associated protocols can be found in the full version of this paper, which is available at: http://tonyrhul.wordpress.com.

Acknowledgement

This work has been sponsored by the NLnet foundation in the Netherlands, when the author worked at University of Twente. The author thanks Arjan Jackmans from University of Twente for his helpful discussions.

References

1. M. Beye, A. Jeckmans, Z. Erkin, P. Hartel, R. Lagendijk, and Q. Tang. Literature overview - privacy in online social networks. Technical report, Centre for Telematics and Information Technology, University of Twente, 2010.
2. C. Boyd and A. Mathuria. *Protocols for Authentication and Key Establishment*. Springer, 2004.
3. Y. Dodis, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 523–540, 2004.
4. M. Hay, G. Miklau, D. Jensen, D. Towsley, and P. Weis. Resisting structural re-identification in anonymized social networks. *Proc. VLDB Endow.*, 1(1):102–114, 2008.
5. A. Jeckmans, Q. Tang, and P. Hartel. Privacy-preserving profile matching using the social graph. In *The Third International Conference on Computational Aspects of Social Networks*, pages 42–47, 2011.
6. A. Narayanan and V. Shmatikov. De-anonymizing social networks. In *Proceedings of the 2009 30th IEEE Symposium on Security and Privacy*, pages 173–187, 2009.
7. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology – EUROCRYPT 1999*, pages 223–238, 1999.
8. L. Palen and P. Dourish. Unpacking "privacy" for a networked world. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 129–136, 2003.