Mobile Networks and Applications (2021) 26:1281–1299
https://doi.org/10.1007/s11036-021-01761-1

A Framework for Protecting Privacy on Mobile Social Networks

Seyyed Mohammad Safi[1] · Ali Movaghar[2] · Komeil Safikhani Mahmoodzadeh[3]

Accepted: 6 April 2021 / Published online: 29 May 2021

Abstract
In recent years, mobile social networks have largely been developed and gained considerable popularity. An approach to protecting privacy on mobile social networks is the use of encryption and access control. Good alternatives for use on mobile social networks are the Public Broadcast Encryption approach for appropriate concordance and consistency with the structure of social networks as well as the Attribute-Based Encryption owing to its capability and proper implementation of the access control policy. Accordingly, in this paper, a framework was presented based on the Public Broadcast Encryption and Attribute-Based Encryption. Using proxies, we outsourced some of these operations in the proposed framework to reduce the computational load of the end device, accelerate the encryption and decryption operations, and decrease the amount of storage memory for keeping the encryption and decryption parameters required by the users in the end-device. Cloud was also employed to store the shared data and user preferences in the social network. The results of investigating the privacy parameters reveal that our framework is superior to the four compared methods. Additionally, the results indicate that in terms of three important parameters in mobile social networks, namely communication, computation, and storage complexity, our method has less complexity and overhead than they have.

Keywords Privacy · Mobile social network · Broadcast encryption · Access control · Outsourcing · Cloud

1 Introduction

With the evolution of smartphones, tablets and other mobile communication tools enabling communication at any time and place, social media is increasingly influencing human life. Nowadays, privacy is one of the most important issues related to social networks, since the conditions prevailing such networks expose them to the violation of the users' privacy and disclosure of personal information. There are various definitions for privacy. Alan Westin defines privacy as "the right of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others" [1].Chuck Kroff and Chalet Trebock define privacy as "the right of individuals for their ranked control and deleting personal information at their own discretion and at a controlled rate of disclosure of personal information" [2].

Gross et al. divided social network privacy into three general categories, namely monitoring and privacy, social privacy, and organizational privacy [3].People who are sensitive to the protection of their privacy, initially refuse to join social networks; however, because of the attractions of social networks and the presence of their friends in these networks, they are encouraged to join [4].

Nowadays, there exist various social networks; Facebook is one of the leading social networks with more than a billion members now. Other companies, such as WhatsApp, Telegram, Twitter, and Google, are also quite active and support millions of users, with all their applications available to smartphones. Usually, the shared information remains on the servers of the social networks for a long time; however, the data may be deleted by the user. Social network providers have full control over the user information [5]. Such users and their data can be a target for privacy violations. Krishnamurthy et al. [6] showed that personal identity

✉ Seyyed Mohammad Safi
msafi@cs.sharif.edu

Ali Movaghar
movaghar@sharif.edu

Komeil Safikhani Mahmoodzadeh
komeil.safikhani@yahoo.com

1 Faculty of Science and Engineering, International Campus-Kish Island, Sharif University of Technology (SUT), Tehran, Iran

2 Department of Computer Engineering, Sharif University of Technology (SUT), Tehran, Iran

3 Department of Computer engineering, Ahvaz, Branch, I.A University, Ahvaz, Iran

information could be leaked by different social networking services. In this regard, some important and relevant requirements pertaining to the concept of privacy and services needed by users are as follows [1, 7, 8]: Privacy, Integrity, Availability, Access control, Fairness, Communication privacy, Auditing, and Validation. Samia Oukemeni et al. [9] fully categorized the types of approaches to protecting privacy and reducing threats as follows:

- **Anonymity:** User data on social networks are a good source for good intention research purposes as well as the main target of profiteers.
- **Decentralization:** Decentralized architecture is considered a solution to reduce the access level of social network providers to user data.
- **Encryption:** Not only does encryption confidentially store the user data on a social network storage server, it is not intercepted when transmitted or received on the social network.
- **Information security:** This concept is not separate from privacy; the security measures taken by users ensure that published data are accurate, and spam, fake profiles, phishing and other threats cannot harm users.
- **Fine-grained privacy and access control settings:** The privacy settings can vary for each individual depending on users' behavior. Considering users' different communication qualities, the details of their access control also vary.
- **User awareness and behavior change:** Users' increased awareness of privacy and threatening behavior changes such as the Letter of Commitment and Adjustments causes the social network to protect and enhance privacy.

In this regard, both encryption and access control will protect the users' privacy on social networks. The existing encryption methods are utilized in most of the proposed methods to protect the privacy, but they differ in how they are executed, implemented, and managed. It is attempted to choose appropriate policies and settings to employ encryption methods according to the characteristics of mobile social networks. Managing and building the key, as well as retrieving and assigning them to users, are some of the challenges of existing methods. Meanwhile, process and computation of volume in encryption, decryption of user messages, as well as storage of each user's encryption parameters are other important issues. In social networks, regarding the types and extent of communication among users, it is essential to utilize a method that can support all types of these modes. Nevertheless, most existing methods do not support the many-to-many mode. Here, all of the above privacy requirements and concerns to implement and execute encryption as well as the access control for mobile social networks are addressed. In this paper, by providing a suitable framework for mobile social networks, we attempted to meet all the privacy requirements and users' needs.

The main contributions of this paper are: (1) We proposed a many-to-many encryption method on mobile social networks, in which in addition to high safety, it contains low computation, communication, and storage complexity. This method optimally and safely performs to outsource the cryptographic operations in order to increase the speed and computational power, and to decrease the need for the storage space. (2) Our method provides a safe framework by combining the features and strengths of attribute-based encryption in the public broadcast encryption. (3) In the proposed method, management and protection privacy will be considered user-oriented, so that it can prepare the social network while others are not able to violate the users' privacy. The investigation results demonstrate that the proposed method has less complexity and overhead than the four similar methods have.

The present paper has the following structure. Section 2 reviews the related previous works. Section 3 investigates the proposed framework and system in detail and discusses the manner in which privacy is achieved. Section 4 describes how to implement the mobile social network proposed by our framework. Section 5 analyzes the privacy and evaluates the efficiency of the proposed system, and finally Section 6 concludes the discussion.

2 Previous works

Several implementations and requirements have been proposed to protect the users' privacy on social networks, so that various methods, including anonymity methods have been provided. The main problem with data broadcasting is the disclosure of information for which anonymity is a solution. Anonymity methods comprise k-anonymity [10, 11], and social network graph manipulation by adding and removing edges and nodes [12], and injecting uncertainty [13].

In this regard, the decentralized architecture is another way that in some ways, in addition to preventing access by unauthorized users, protects the users' privacy by decreasing the access level of social network providers to the users' personal information. Some of these methods include Persona [14], EASiER [15], Lockr [16], flyByNight [17], Face Cloak [18], NOYB [19], and CP2 [20]. However, centralized architecture also offers effective methods to protect users' privacy. For example, Xiao et al. [21] developed a method with a centralized architecture that is effective in protecting users' privacy, particularly in relationships with friends and users' location.

Other studies indicate that increasing user awareness leads to protection and enhancement of their privacy [22]. The level of privacy in MSNs depends on the applications, the point of view (sender and recipient), and the trust level between entities [23]. Mechanisms of sending and the types of data may be threats to privacy [24].

However, the use of encryption is well known as one of the most widely used methods to protect users' privacy on social networks. Hence, considerable research into privacy is based on the encryption reviewed in the following. Relying on mathematics, encryption methods have constantly evolved over time. Tootoonchian et al. [16] presented a method called Lockr, to gain access control to users' shared data. In this method, the user issues a social certificate to each of his friends and if he wants to share data with them, he assigns a list of friends with the certificate to the data. Lucas et al. [17] proposed a method called flyByNight in which a second server provides proxy encryption.

In the method proposed by LUO et al. [18], known as Face Cloak, users deceive social network servers by storing fake data on the server instead of their own shared data; there is another server in addition to the serving server on which the shared data are stored while encrypted. In this method, there is no appropriate access control. Guha et al. proposed NOYB whose most important achievement is protecting the privacy of the user data against the social network provider and non-friend users. This method has done nothing to establish access control. Song Ling Fu et al. [25] named their approach Cadros, which aims at improving data access in a decentralized social network with session privacy. To achieve privacy in Cadros, the erasure coding technique is used to store data on the cloud server, and data are stored in the circle of friends using complete duplication.

Kai He et al. [26] proposed the Identify Based Broadcast Encryption (IBBE) scheme, which is able to provide complete confidentiality and anonymity for encrypted text against various attacks. However, considering the high encryption time, the scheme is not particularly suitable for social networks with a large number of users. Schillinger et al. [27] developed an end-to-end encryption approach that is true for online social networks; it focuses more on messaging and online chat services and does not consider other users' needs in social networks. They proposed an end-to-end encryption approach using the RSA public key encryption algorithm and the AES symmetric key encryption algorithm. Ahmed Khalil Abdulla et al. [28] developed a flexible system providing security and privacy based on encryption-based access control. This method, called *Hide* in The Crowd (HITC), allows users to post data on their social network platforms based on access control with the appropriate grain level and decrypts data for the target users. HITC is designed as a browser extension and can connect to any existing OSN platform without the need for a third-party server using the RSA encryption.

Shamir et al. proposed a method to share secrets [29], called threshold scheme (k, n), in which the secret D is divided into k fragments. Rebuilding the secret is possible if each k piece or more is available. Later, Shamir et al. introduced a new encryption scheme called Identity-Based Encryption (IBE) [30]. This method allows users to safely communicate with each other without public key exchange, using the recipient's unique identity like IP address. Without the key exchange, the risk of key disclosure is minimized. Sahai and Waters proposed the first ABE scheme [31], which uses IBE and the idea of secret sharing to generate interval access control. ABE perfectly combines access control and encryption [32]. ABE also facilitates key management by utilizing attributes instead of encryption components [33]. Subsequently, the ABE fine-grained access control scheme was raised by Goyal [34] and Benthencourd [35], adding the fine-grained property of access control to attribute-based encryption. The Persona method proposed by Baden et al. [14] has an important attribute, which is the ability of the user to divide friends into different groups; like many methods, this approach uses encryption to maintain privacy. The encryption technique is ABE, which is based on attribute.

Jahid et al. [15] applied changes to the attribute-based encryption system presented in the Persona method in the EASiER method. Identical to the previous method of shared data, this novel method further provides users with attribute-based encryption (ABE). This method allows the user to regenerate the new proxy key and encrypt the data by removing the intended attribute from the user or favorite users if he wants to remove the attribute from his friend. This ultimately prevents the deleted users from accessing the new data. Addition of friend users to the group functions in the same way, which is not seen in Persona, is still limited, and the combination of attributes is impossible for users. It also uses a second party (other than a social network server) to solve the problem of inflexibility and non-dynamicity of decryption in access control in the previous method. In other words, EASiER improves access control and changes the group (adding and removing users) without disrupting access control.

Despite their considerable benefits, group encryption methods are unsuitable for use in the real world owing to the computation burden or the large size of the key. Therefore, Sun et al. [36] proposed a method to encrypt outsourcing, which operates based on attribute-based encryption. Their design suggests an external source-decrypting algorithm and the comparison of CP-ABE on the cloud-based mobile social network. In this design, the comparison phase is added prior to the decryption, and the proxy-decrypting task is performed mostly by the representation of the user.

As the name implies, broadcast encryption is used to safely broadcast information to a group of people, so that outsiders cannot decrypt information even by collusion, which was first developed by Fiat and Naor [37]. Broadcast encryption was eventually evolved by Boneh et al. using bilinear maps, where the BDHE-based method is developed with sufficient security and constant-size encrypted text. A few years later, Gentry and Waters, who were Boneh's colleagues in the previous scheme, strengthened the security of the previous method to reinforce the proposed broadcast encryption method against static and adaptive attacks. However, the drawback of this method is the dependence of the message size on the number of group members, which is due to addition of the ability to prevent adaptive attacks. It should be noted that as in the previous method, this supplementary model is also resistant to the collusion of group members [38] [39] [40].

In the following, many changes are made to this method. One of these changes, allowing the user to freely add and remove friends, was made by Malek and Miri [41], which is referred to as ASBE (Adaptively Secure Broadcast Encryption) in the present paper. In this encryption method, the user performs the encryption with his key for a group of users. Moreover, only the addressed users can decrypt data with their own private key, the parameters contained in the public key, and the header made by the cryptographer. In this encryption method, CA does not have to change the key by restricting the target audience and the user can simply bypass the key by removing the person from the header. The method proposed by Malek and Miri is delineated in the following. CP2 is another method proposed by Raji et al. [20]. This method uses public-key broadcast encryption. The scheme attempts either to entirely abandon the social network server or completely prevent interrupting the encryption and key distribution processes; instead, it uses a second partner as a proxy, and as semi-trusted, or uses the server as semi-trusted, and the user himself undertakes the tasks related to encryption. Here, this encryption method is used to make the data broadcasting more flexible and dynamic while protecting privacy in access control.

In [42], Safi and Safikhani compared two methods of broadcast encryption and attribute-based encryption, to some results presented in the following. In the scheme proposed by Malek and Miri [41], the whole system needs to be re-setup if a new user wants to join the system; therefore, the size of the key, the encrypted text, and the time elapsed increase linearly. Resetting up the system is costly and useless in the real world. Scalability in the ABE system allows a user with new attributes to join without the need to re-setup the system. In other words, the system can dynamically expand; however, there is no such possibility in broadcast encryption. Unlike attribute-based encryption, broadcast encryption does not have the key revocation issue, which is still the unresolved problem in ABE. The only proposed solution to this problem without the need to replace the keys is to use the time attribute to generate a key that is periodically replaced. After a person has been removed, they are not authorized to use this key. A problem with this key is the added encryption burden [43]. In broadcast encryption, determining those who can access the data depends on their consideration of the built header; thus, it is easy to bypass the encrypted file. However, broadcast encryption lacks the good attribute-based encryption property, meaning fine-grained and attribute-based access control. Gao et al. proposed a method to outsource and store the user data in cloud based on multiple keys in mobile social networks protecting users' privacy. They used a proxy re-encryption scheme with additive homomorphism [44]. Ali et al. provided a framework to share users' content on social networks, which ensures users' security and privacy. They have attempted to address the concerns of data owners and co-owners to manipulate, collect or abuse the data by unauthorized users. In this framework, they have used cryptography and access management to protect privacy and security [45].

In this paper, a framework was proposed to protect the privacy of mobile social networks by using the public broadcast encryption presented by Malek and Miri [41], modifying the framework, modeling the attribute-based encryption, and adding its capabilities to the broadcast encryption. In our framework, the used proxy, while reducing the computational burden, makes key collusion impossible for unauthorized users. The proxy is also able to maintain the encryption keys, and part of its own decryption keys assists the user in building the header and session key, which is explained later. Due to the limitations of ASBE, all possible members must be predicted beforehand, creating the need for a very large multiplication group. This enlarges the size of the keys and increases the communication overhead in building keys encompassing all members; however, they never communicate with one another. It is also difficult to predict the number of people who will join the social network and the number of times they will request a new key. Furthermore, if the number of keys made for them is high, even if they are not attributed to anyone, it will cause computational overhead and storage, which has been addressed in our proposed system.

3 The proposed framework

This paper proposes a framework for mobile social networks, in which to protect privacy by using proxy and cloud space, different conditions are provided for smartphone users to easily conduct processes related to this framework, including encryption, decryption and storage. In this framework, the user can determine certain attributes for the users of the broadcasted data and apply certain access controls to share their content. Our proposed framework presents a system, called single-layer system, with two different implementation methods. In this system, the attributes that the user identifies for their friends to access their shared data, are all applied together in the header for encryption and decryption. The system is described below. As Fig. 1 shows, the system model consists of five sections: Data Owner (DO), Cloud Server (CS), Certification Authority (CA), Data Users, and Proxy.

The data owner defines access policies, encrypts data under access policies, generates the header required for decryption with a proxy-assisted session key, and assigns the encrypted

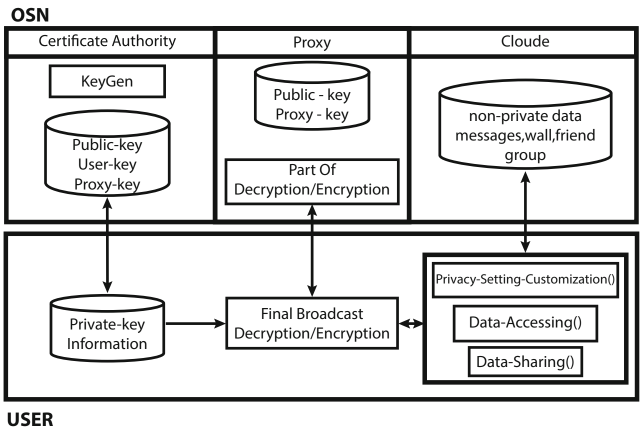

Fig. 1 The proposed framework architecture

text to cloud servers, along with the header. To read the encrypted text, the users requesting shared content, first call it from the cloud with the header, request the key from the proxy, and start the decryption if the key is approved and given by the proxy. Cloud is where the data are stored whether the shared data or the users' parameter is not needed by anyone except users. This storage enhances users' freedom of action and provides control over their data. The Key Operator is a fully trusted organization responsible for generating attribute keys for users and distributing them at the request of users and their attributes. This section is related to the proxy, and for each key it generates for the user, it also gives a key to proxy to assist the user in encrypting and decrypting. In our proposed framework, CA is considered outside the social network provider's control.

In this framework, the proxy is semi-trusted, meaning it is reliable enough to perform half of the key generation computations of the header on behalf of the user, and users' collusion can also be prevented. The proxy does half of the computing in the key acquisition process, reducing the amount of encryption and decryption work on the user's side. Note that the key provided by the proxy operator does not enable the proxy to decrypt information in any way. Since the proxy-generated key belongs to its target user, no one else is able to use it. This is due to the user-specific random variable, which is eventually eliminated in the computations by the second key application by the user; this is explained below. This attribute ensures that the key transfer to the user does not require a secure connection, since in the event of an eavesdropping or attacks on this basis, considering the lack of a random variable chosen by the user, it is impossible to complete and generate the final key; therefore, the secure path is unnecessary. Note that each user can have their own proxy. The necessity of the user's choice of the proxy is to report it to the key operator, and other users do not need to know each other's proxies. However, we consider only one proxy for all users in the system. Figure 1 shows the overview of the proposed system and how the components interact with each other.

In the proposed method, the CA separately encrypts and decrypts the attributes and provides the users with them. Attribute keys are individually generated for the attributes intended by the user. These keys represent the shares putting users in a group (for example family and friends). Nevertheless, only one key with one general attribute like membership can be defined in the network, and there is no need to allocate multiple attributes. The encrypting user initially encrypts the data with the attributes performing the policy of access control based on the access control policy. Next, they share it in the cloud storage for users with the existing attributes. After generating keys for people with different attributes, the CA divides the key into two parts, one assigned to the proxy and the other to the user. The difference between the two methods is in assigning each part to a proxy or user, each with its own advantages. In the broadcast encryption, the encrypting user needs to generate a header and send it, along with the encrypted text, so that the header allows the authorized users to decrypt it. This header is provided for the proxy using the encryption keys and the random number selected by the encryption user is then generated. In the header-generation process, the proxy helps the encryption user to reduce computations and immediately consider those with attributes. In this regard, the encrypting user assigns his encryption keys to the proxy to generate the header of each attribute, and the proxy generates the pseudo-header according to the people existing in that attribute, returning it to the encrypting user by completing it. These headers and keys vary depending on each attribute. Afterward, the user respectively encrypts the data based on their intended preferences and the access control policy, and places it in the cloud, along with the headers. The structure of our system consists of four phases: system startup, key generation, encryption and decryption. The functions of our proposed system are similar to those presented in the broadcast encryption presented in [41]. The difference is that in addition to the variations, the implementation of functions and the management of keys are different, and the attribute-based encryption [43] is further added.

3.1 Single-layer system

In this system, the user first determines all his intended attributes based on the access control policy and inserts them in a header. If the user is dissuaded to allocate one or more attributes to one or more friends, it will be easily possible by modifying the list of authorized persons, and there is no need to regenerate the key. Each system has four operational phases, namely system startup, key generation, encryption and decryption. Both methods are described below.

3.1.1 The first method

Below are the four phases that make up the system.

Phase 1: System setup.

CA is responsible for generating public keys, private keys, and the parameters required to generate them. Individual private keys, encryption and decryption keys are generated by CA using multiplicative groups and bilinear maps. To setup the system, per each attribute (s), the setup function receives the security parameter and the maximum number of possible members (n) as the input, generating the master key and the public key. For this purpose, it generates multiplicative groups and bilinear maps (G_0, G_1, e) and random values α, ϒ from Z_P^* that will be different for each attribute or set of attributes and selects a group of generators like g∈ G_0. The keys generated for each attribute are as follows: Unlike [41], the public key here is divided into two parts, which are explained in the following.

$$PubK_1 = \left\{ \mathrm{G0}, \mathrm{G1}, \mathrm{e}, \mathrm{g}, e(\mathrm{g},\mathrm{g})^{\alpha^{2n+1}} \right\} \tag{1}$$

$$PubK_2 = \left\{ \left\{ g^{\alpha^i}, \forall \mathrm{i} \in [0, 2\mathrm{n}] \right\}, \{x_1, \ldots, x_n\} \right\} \tag{2}$$

$$SK = (\alpha, \Upsilon) \tag{3}$$

Phase 2: Key generation.

The KeyGen (i, SK) function generates member keys for each user with the index i by the master secret key. To this aim, a r_i is selected for each member and for all attributes, and is proportionate to the target members able to decrypt the data, a B_j is randomly selected from Z_P^* for all attributes. The private keys of a property for each member are as follows:

$$PrVK_i = \left\{ r_i, d_i, T_{i.j} \right\} \tag{4}$$

$$d_{i,\mathrm{j}} = g^{\Upsilon \alpha^n B_j \frac{\alpha^2 - r_i}{\alpha - i}} \ (\forall \mathrm{j} \in [1, \mathrm{n}], \mathrm{j} \neq \mathrm{i}) \tag{5}$$

$$T_{i,\mathrm{j}} = g^{\frac{\alpha^i}{\gamma B_i}} (\forall j \in [0, n]) \tag{6}$$

$T_{t,j}$ includes the n + 1 private keys used to generate the broadcast data for the n-1 member of the system with that attribute, and $d_{i,j}$ includes other n-1 private keys utilized to decrypt the broadcast data received from the n-1 system member, along with r_i. In the proposed approach, CA assigns all $T_{i,j}$ to the proxy for use in the header process. $PubK_1$ is further sent to the proxy and users to use for encryption and decryption.

$$key_1 = \{r_i, PubK_2\} \tag{7}$$

$$key_2 = \left\{ d_{i.j} \right\} \tag{8}$$

Each set of key_1 and key_2 can only be used for one pairing to obtain the final symmetric key.

The values sent to the user and the proxy are as follows in the first method:

$$user = \{PubK_1, key_1\} \tag{9}$$

$$proxy = \{PubK_1, key_2\} \tag{10}$$

Phase 3: Encryption

At this phase, data owners specify access control to access their data. By specifying the attributes for each intended attribute, the data owner with the index i selects a random t from Zp* and separately computes the DEK value for each attribute as follows:

$$DEK = e(g, g)^{t\alpha^{2n+1}} \tag{11}$$

Then, it sends the $C_2^{'}$ generation request to the proxy for the members in that attribute. The proxy multiplies the $T_{i,j}$ of each user's attribute based on the j member of the attribute to obtain $C_2^{'}$ of that attribute. The header is then generated by t and $C_2^{'}$ as follows:

$$C_1 = g^t \tag{12}$$

$$C_2^{'} = g^{\frac{\sum_{j=0}^{n} e_j \alpha_A^j}{\Upsilon_A B_i}} \tag{13}$$

$$C_2 = C_2^{'\,t} = g^{t \frac{\sum_{j=0}^{n} e_j \alpha_A^j}{\Upsilon_A B_i}} \tag{14}$$

Finally, the header is obtained as follows:

$$Hdr = \langle C_1, C_2 \rangle \tag{15}$$

Note that DEK is utilized to encrypt the broadcast data, and the header carries the data required for decryption by authorized members. Data encrypted by DEK is uploaded in the cloud, along with the header and access control policy for users' access.

Phase 4: Decryption

During decrypting, the user with the i-index primarily calls the encrypted text from the cloud and requests the required keys from the proxy according to the access control policy with which the data are encrypted and sends the C_1 value to the proxy. The proxy checks the membership or non-membership of the user i in all the requested attributes, and if everything is true, using its own keys, the proxy computes the first required bilinear map by the following values:

$$p_i(\alpha) = \alpha^{2n+1} - \frac{\alpha^{n+2} p(\alpha)}{\alpha - i} \tag{16}$$

$$h_i(\alpha) = \alpha^n \frac{p(\alpha)}{\alpha - i} \tag{17}$$

The value of DEK' is calculated as follows and sent to the user.

$$DEK' = e\left(C_1, g^{p_i(\alpha)}, g^{r_i h_i(\alpha)}\right) \tag{18}$$

Having received DEK', the user computes DEK as follows:

$$DEK = (DEK')e\left(C_2, d_{i.j}\right) \tag{19}$$

The above equation generates the encryption key and the user can perform the decryption operation. Note that the DEK' key is not used by any other member; therefore, the key collusion operation is prevented while reducing the computation.

$$\begin{aligned} DEK &= e\left(C_1.g^{p_i(\alpha)}, g^{r_i h_i(\alpha)}\right)e\left(C_2, d_{i.j}\right) \\ &= e\left(g^t, g^{p_i(\alpha)+r_i h_i(\alpha)}\right) \\ &\quad \times e\left(g^{t\left(\gamma B_j\right)^{-1} p(\alpha)}, g^{\gamma\alpha^n B_j \frac{\alpha^2 - r_i}{\alpha - x_i}}\right) \\ &= e(g,g)^{t(p_i(\alpha)+r_i h_i(\alpha))} e(g,g)^{t\alpha^n p(\alpha)\frac{\alpha^2-r_i}{\alpha-x_i}} \\ &= e(g,g)^{t\alpha^n\left(\alpha^{n+1} - \frac{\alpha^2 p(\alpha)}{\alpha - x_i} + r_i\frac{p(\alpha)}{\alpha-x_i} + p(\alpha)\frac{\alpha^2-r_i}{\alpha-x_i}\right)} \\ &= e(g,g)^{t\alpha^{2n+1}} \end{aligned} \tag{20}$$

3.1.2 The second method

The four operational phases are as follows:

Phase 1: This phase operates like the first method.
Phase 2: The only difference in Phase 2 is the keys sent to the proxy and the user, which are as follows:

$$user = \{PubK_1, key_2\} \tag{21}$$

$$proxy = \{PubK_1, key_1\} \tag{22}$$

Phase 3: It is exactly the same as the previous method.

Phase 4: During decrypting, the user with the i-index primarily calls the encrypted text from the cloud and according to the access control policy with which the data are encrypted, requests the proxy for the required keys and sends the C_2 value to the proxy.

As in the previous method, the proxy checks whether or not the user i is a member; if everything is true, using its own keys, the proxy computes the second required bilinear map using $d_{i,j}$ in all the requested attributes. The DEK' value is calculated as follows and sent to the user.

$$DEK' = e\left(C_2.d_{i.j}\right) \tag{23}$$

Having received DEK', the user computes DEK using the formulas $p_i(\alpha)$ *and* $h_i(\alpha)$ as follows:

$$DEK = (DEK')e\left(C_1, g^{p_i(\alpha)}, g^{r_i h_i(\alpha)}\right) \tag{24}$$

The accuracy of the key is guaranteed as in eq. 20. The result of the above equation is also the key to the desired broadcast encryption. The advantages of the first method is that the user is not required to calculate $p_i(\alpha)$ and $h_i(\alpha)$. Rather, they can always decrypt those attributes with the key values of $d_{i,j}$ attributes. In this method, the proxy has to obtain the large public key required for decrypting, which is different for each sending user. Accordingly, the space occupied by the end-device is reduced, and the phone's limited processing resources are not used. The advantage of the second method is that the user only needs one r_i for all the existing attributes, resulting in reduced numbers of stored keys regardless of the number of attributes; however, the user still requires a public key that can be sent with the encrypted file every time. Therefore, storage is no longer an issue. In systems where there is no concern about the communication cost and the main issue is the storage space, this method is highly efficient as it only requires one key (e.g. 512 bits) for all attributes.

4 Implementation of the proposed framework

This section includes algorithms for mobile social network setup operations, user session with mobile social network, mobile social network login, personalization of privacy settings, data sharing, data access and logging out of social network. The following shows how to implement the proposed framework.

4.1 Setting up Mobile social network

Figure 2 is a pseudo-code of the mobile social network. First, the social network auxiliary components are set up to provide the necessary background for users to join the social network (Lines 3 and 4).

Setting up auxiliary components: At this stage, a storage server capable of reading, writing and searching data is considered. A proxy called Encryption Proxy also helps users with encryption and decryption (Line 8). In the cloud, in

```
01 MSN-setup ()
02 Begin
03      Entities-Setup ()
04      User-Setup ()
05 End
06 Entities-Setup ()
07 Begin
08      EncryptionProxy-Setup ()
09      Cloud-Setup ()
10      KeyGen-Setup ()
11 End
//Certificate Authority Section
12 KeyGen-Setup ()
13 Begin
14      (PubP, MPK, SK) =PBE-Setup ()
15 End
16 User-Setup ()
17 End
//User Section
18 Begin
19      PhoneNumber=Enter-PhoneNumber ()
20      CheckPhoneGenuinity(SmsSentCode==SmsRecivedCode)
21      UserInfo=Get-Information ()
22      Send-Cloud (PhoneNumber ,UserInfo)
23 End
```

Fig. 2 Setting up mobile social network

addition to data storage, the list of users is included to simplify searching for friends (Line 9).

User registration: When a user wants to join a social network, he enters a mobile number (Line 19). Next, he enters other information such as name and other attributes to be found by friends in searches (Line 20). This information is then sent to the cloud and stored there (Line 21). Furthermore, whenever the CA wants to send the key to users or proxies in the response of users' requests, the authenticity of the user will be checked.

4.2 User session with mobile social network

Figure 3 shows the pseudo-code series required at the user meeting with the social network. In the first step, the user prepares his current environment (Line 3). Afterward, he performs the three stages of privacy setting customization, data sharing and data access based on his needs (Lines 7, 8 and 9). At the end of the session, the operation of the completion of the session is done between users and the mobile social network (Line 11).

```
01 User-Session ()
02 Begin
03      Login ()
04      Loop
05           Begin
06           Customize-Privacy ()
07           Share-Data ()
08           Access-Data ()
09           End
10      Logout ()
11 End
```

Fig. 3 User session

4.3 Logging in to mobile social network

With each login to the mobile social network, the user prepares the information required for social interaction in the current session. This information is stored in the cloud prior to leaving the social network. Figure 4 shows the pseudo-code.

Information like the mobile number is publicly stored in the cloud, so that one's friends can find it and become aware of their membership in the social network. Other information such as new attributes given to the user and announcement of a new message will be encrypted in a part of the user interface called Wall by friends to inform the user. Private information like communications can also be stored in the cloud in an encrypted manner for use when needed. Since mobiles themselves have memory, storing information to the cloud is unnecessary, and even if the information is erased, it can always be regenerated, and the key can be requested by the key operator.

Logging into the social network is a simple process. As Fig. 4 displays, the user first reads the information on their wall to become aware of the requests and messages. In this regard, by removing the keys, they recover the keys stored from the previous decrypts from the memory. (Line 3) Subsequently, they decrypt the information on the wall (Line 4). Information on the wall can be used by user-generated keys in previous communications to retrieve symmetrically encrypted content that will not incur cost to the user to decrypt.

4.4 Customizing privacy settings

Figure 5 presents the pseudo-code of the process of privacy setting personalization. At this stage, social relationships and attributes are simulated in the social network. This pseudo-code explains how to use the cloud and Encryption Proxy.

The user can add and remove friends (Lines 3 to 6) and further define attributes to control access, or assign attributes to friends or take these attributes back from them (Lines 7 to 10). To add friends in the suggested method, the user is able to initiate relationships with other users at any time. As soon as the users log into the social network, they can search for their friends' profiles in the proxy, which is the mobile number here (Line 15), and send a friend request if the person is on the social network (Lines 16 and 17). When the user accepts the request, he adds the requesting person to his friend list and

```
01 Login ()
02 Begin
03           UsedKeys=Retrieve(UsedKeys)
04           Decrypt (Wall ,UsedKeys)
05 End
```

Fig. 4 User logging

```
01 Customize-Privacy ()
02      Begin
03           if (there is friend to be added)
04                Add-Friend ()
05           if (there is friend to be revoked)
06                Remove-Friend ()
07           if (there is any changes to be make on attribute)
08                Change-Attribute ()
09      End
10 Add-Friend (PhoneNumber)
11      Begin
//user section
12           Search-Cloud(PhoneNumber)
13           if (friend is a member of MSN)
14                send-request (PhoneNumber)
//friend section
15      if (accepting friend request)
16           Begin
17                add user to the FriendList
18      End
19 Remove-Friend(PhoneNumber)
20           find UserIndexes and remove associated keys
21           send-remove-EncryptionProxy (UserIndexes)
22           remove all information from FriendList
23      End
24 Change-Attribute (Attribute, MaxUserNumber, PhoneNumber)
25      Begin
26           if (assigned to an attribute by other user)
27                Request-Key-CA (Attribute)
28           if (attribute need to be define)
29                Define-Attribute (Attribute, MaxUserNumber)
30           if (there is a friend to be added in to Attribute)
31                UserIndex = Attribute-Add-User (PhoneNumber, Attribute)
32                Save friend's attribute's index in FriendList
33           if (there is friend to be remove form attribute)
34                remove friend's UserIndex in FriendList and tell EncryptionProxy
35      End
```

Fig. 5 Privacy settings

notifies the person by proxy (Lines 19 to 22). Friendship requests can be made out of bound or with the cloud mediation.

To remove a friend, it is enough for the user to delete the keys for that person and notify the indices associated with the attributes of the person to the encryption proxy, so that he also deletes the person's key and remove him from future encryption headers (Lines 25 to 27). Note that the deletion process is one-way, meaning the deleted person may still share the user data accessible to the user.

Defining attributes The user can assign attributes to his friends. These attributes should be mentioned in clear names, so that other users and friends can further use them to tag their friends and make decryption facile. For each new attribute, a new master key is defined. The largest possible number of members can be assigned by the individual or by the key operator. If the attribute of the data owner is predefined, the data owner only receives his key from the operator (Line 27) who also gives the new user key to the target data owner members so as to decrypt the broadcast data by the data owner. If the new attribute needs to be generated, the user asks the key operator to generate it (Line 29). If the user wishes to attribute the attribute to their friends who added them to his Friend List, the user announces the key number, along with the attribute to the key operator; therefore, in addition to generating the key for the user and the proxy, it generates the target key, sending it to him (Line 31). Afterward, he receives the user index related to the friend's attribute key and stores it in his Friend List to encrypt and decrypt information between himself and his friend (Line 32). If a person wishes to remove a friend from the attribute (with keeping the friendship), the index removes the attribute from the Friend List, discards the relevant encryption and decryption keys, and notifies the encryption proxy. This allows him to further remove the friend from future headings associated with that attribute (Line 34).

As observed, in manner broadcast encryption works, users in one attribute may not be friends and fail to decrypt the non-friend data. This is possible, since the non-friend user is unaware of the mutual decryption key belonging to the encrypting person, and the encrypting person does not put the index of the non-friend user in the header.

4.5 Data sharing

The purpose of creating a social network is to establish interaction via data sharing. Figure 6 presents the proposed pseudo-code. The user initially specifies the attributes in which he wants to share data. The user receives the encryption key and the calculated initial header and applies the key by announcing the attributes to the encryption proxy (Lines 5 to 7). Note that if the attribute members are fixed, the proxy does not need to recalculate the encryption key and the initial header. After receiving the required information, the user makes the final encryption changes to the key and header, encrypting the data

```
01 Share-Data ()
02 Begin
//user section
03      for (each Attributes)
04           Begin
05                PreDec = EncryptionProxy (Attributes)
06                EncryptKeys = KeyMaker (PreDec)
07                Cipher-Text = Encrypt (Data , EncryptKeys)
08           End
09           Upload-Cloud (Cipher-Text, Headers, UserIndexes)
10 End
//EncryptionProxy section
11 EncryptionProxy (Attributes)
12 Begin
13      if (member of Attributes was changed)
14           Begin
15                make PreHeader and PreEcryptionKey associated with users in Attributes
16           End
17           send created data back to user as PreDec
18 End
```

Fig. 6 Data sharing

with the final key (Lines 7 to 9). When receiving a request, the proxy sends the same keys calculated from the previous step if the attribute members are unchanged (Line 17). Otherwise, by making new changes, it resumes the computations and sends the result to the user (Lines 15 and 16).

4.6 Data access

At this point, the user has access to the data shared by his friends. The pseudo-code presented in Fig. 7 shows how the user accesses the data. In this section, the user selects the data informed by the social network proxy and obtains the information required for decryption such as how to apply access control and the attributes used by the index of the encrypting person and the header (Line 3). After investigating the employed attributes, the user reads the encrypted text from the cloud (Line 6) to see whether his index is included in the encryption (Line 6). If the encrypting user changes the employed attribute members, he asks a semi-ready key for decryption in each changed attribute from the encryption proxy by sending a new header (Lines 12 and 13); otherwise, he employs the previous keys. After obtaining the decryption key, the user decrypts the encrypted text (Line 16).

4.7 Exiting social network

At the end of the user session with the social network, the user only needs to store the keys used in the mobile to use them in the next sessions, if they are not changed and use them to decrypt the information placed on the wall. Figure 8 illustrates this process in the pseudo-code.

```
01 Access-Data ()
02 Begin
//friend section
03      Datatype=Select the data type willing to access and check
specified Indexes
04      if (user had the privilege to access data)
05          Begin
06          Cipher-Text=Send-Cloud(Datatype)
07          End
08      if (previous keys was changed)
09      Begin
10          for (each changed keys)
11          Begin
12              PreKey=request-EnProxy (Attribute ,
DOIndex ,Header)
13              ComKey=Process-key(PreKey)
14          End
15      End
16      Plain-Text=Decrypt (PreKeys ,Cipher-Text)
17      End
```

Fig. 7 Data access

```
01 Logout ()
02 Begin
03      Store(UsedKeys)
04 End
```

Fig. 8 Exiting social network

5 Investigating the privacy and performance evaluation

In the framework presented to evaluate the privacy, we investigate the three sections of users, social network provider, and CA. We attempt to help users in the process of using the privacy service considering all the limitations of the smartphone by using the cloud space and proxy. Only users who obtain the necessary CA permissions following authentication are able to work on social networks. Users employ proxy computation capacity for encryption and decryption operations without allowing the proxy to access the user data. Proxies serve users as concerns the generation of a part of the encryption and decryption keys that perform high computational work and, but can never obtain the complete encryption and decryption keys. The other part of the key is under the user's control, and he performs the generation of the end key; therefore, as the last encryption and decryption loop and access control, the user plays a key role in privacy. Since CA is independent of its intended social network provider, it is also unable to access its users' data. Moreover, the social network provider that interacts with and synchronizes between the system components is unable to obtain the encryption and decryption keys even by colluding with other components. Cloud also only receives data that users decide to store and share. Since they have no key to decrypt, they are unable to access the data. In this framework, if the user only intends to communicate with others and send a message of any kind, he is not limited considering the session process of the user and key generation and can easily receive a message in a secure environment. Privacy needs are provided in a way not interfering with users' online and fast access. Accordingly, only the phone numbers of people on the social network are visible, and other user information is protected. Each user is identified by this phone number and ID, which is only possible for the user and his friends. Moreover, the relationship between the phone number and the ID of the user is hidden from the CA and proxy; even the user's Friend List remains hidden from others.

Considering the flexibility of the method described in the previous section, it is easy to modify the list of authorized users by the data owner. Backward and forward confidentiality is also fully guaranteed in access control. In addition, it is possible to define groups, change their members, and update the terms and attributes and any combination of attributes to grant access to the shared data. The user can only grant access to data to specific people or to members of a group, meaning any group and subgroup can be created without limitation.

Adding and removing new users to the user friend list, forming and deleting groups, deleting contact users, and merging groups can be easily performed by changing the attributes and keys. For this reason, any arrangement of users' preferences can be provided to authorized people to access their shared data at the lowest cost.

Within the provided framework, real-time and appropriate cooperation of CA and proxy is essential. Furthermore, users must use the same cloud. If they want to use several cloud environments, they must all be introduced to their target audience. In this regard, both selection and management of cloud, CA and proxy are users' responsibility. Here, proxy is considered semi-trusted.

The proposed framework fully encompasses the privacy requirements of a social network using the encryption and the access control it provides.

5.1 The performance

In this section, we compare our proposed method to ASBE, EASiER, CP-ABE and CP2 encryption methods based on Facebook social network parameters. The reason why these methods were chosen is the proximity of their performance in securing their privacy to the proposed framework. The study comprises three important factors, namely communication complexity, computation complexity, and storage complexity.

To calculate the computation complexity, by calculating the running time of pairing, point multiplication (MUL), and point exponentiation (EXP) operations, we have developed a software program running on a Samsung Galaxy Note 8 with an 8-core processor, 6GB RAM, and Android 9.

On Facebook, people can define 300 groups with members growing up to 1000 people. These different groups are defined based on the difference in the characteristics of the individuals in these groups. Furthermore, anyone on Facebook can be friends with up to 5000 people. According to [46], each person on Facebook makes an average of 150 friends and is a member in 85 groups. Similar to Facebook, our proposed framework considers the above capabilities. In our scenario, a maximum number of 300 and a mean number of 150 attributes are defined. Obviously, there is no limit to increasing the number of attributes. The maximum number of users' friends is 5000 and the average is 150. In our scenario, each new user can add a maximum number of friends, considered *n*, added to the Friend List. Additionally, any user can delete any number of friends. Each user can define the maximum number of groups, called N. Each group can have a maximum of L members. We use y to represent the maximum number of attributes a group can have, and N/2 represents the mean of attributes. In our evaluation, definable values and average values are considered based on Facebook.

In the following, three factors of communication complexity, computation complexity and storage complexity are evaluated. For simplicity in the figures and tables, the first method is called N1, and the second N2.

5.1.1 The complexity of communication in the single-layer system

This section calculates the communication overhead. These computations are investigated in the processes of deleting and adding friends, as well as sharing and accessing data for one attribute, the results of which can be generalized to multiple attributes. This investigation focuses on the number of messages to be exchanged for different operations required by each social network user.

Since the structure of broadcast encryption [41] is based on the elliptic curve group elements and bilinear maps, these studies are conducted on this basis only by considering the elliptic curve elements. The size of these elements is considered 512 bits here. It is noticeable that the costs in which the users are involved are considered.

Let us investigate the communication complexity by examining the five modes with the most interactions between the user and the server of the social network and its components. These modes, the communication complexity of which is examined, include setup, adding friends, data sharing, removing friends, and accessing data. Table 1 compares these modes in different methods.

The data owner is responsible for the setup stage, the EASiER method, and the CP2 of all computation operations, removing the need to transfer and communicate between the user and the social network server to send and receive data; therefore, no complexity is needed. In CP-ABE and N2, only three multiplicative elements required for future encryption processes are generated. These values are sent to users by a CA or social network server; thus, the communication complexity is limited to these three elements of the multiplicative group. In ASBE and N1 methods, the public key generated with the pair e (g, g) is generated and sent by the CA to the user, following the communication complexity of 2n + 2. The N2 method operates almost like EASiER, CP-ABE and CP2; however, in N1 and ASBE methods, the exchanged multiplicative group elements are more, since the keys are received from CA. To add a friend in the ASBE method, friends must receive two keys, namely decrypting and encrypting keys related to those friends. If n is the maximum of friends, the 2n key must be distributed among friends. This value is reduced in N1 due to the need for only one decryption key for one friend; other keys are maintained in the proxy for each decryption key, so that one key is required for each person; therefore, instead of 2n, there is a need for n keys, since the other n is available to the proxy. In the N2 method, since the decryption and encryption key is in the proxy, adding a friend to the Friend List suffices, and the proxy is to be notified. The proxies are responsible for adding friends. Note that the informing

Table 1 Communication complexity

	Setup	Adding friends	Data sharing	Removing friend	Data accessing
EASiER	0	n(3 N/2+1)	2y+2	2y+2	2(3y+2)
CP-ABE	3	n(N+1)	2y+2	2y+2	2(2y+2)
ASBE	2n+2	2n	2	2	4
CP2	0	n	2	2	4
SL-N1	3	n	2	2	4
SL-N2	2n+2	1	(2n+4)	2	(2n+4)

proxy is handled by an identifier reaching the $\log_2 n$ size. For instance, with a 2-byte ID, 65536 people can be assigned IDs. Given the small amount of the two bytes of the 512-bit elements considered in the communication complexity, we decided to ignore them.

To share data among users in ASBE, CP2 and N1 methods based on broadcast encryption, the generated header includes only two elements of the multiplicative group; in the N2 method, the public key values, equal to (2n + 4), are further added. However, in EASiER and CP-ABE methods, these values are calculated based on the number of attributes and the need to receive the CT parameter [15], equaling 2y + 2.

Removing a friend in ASBE, CP2, N1 and N2 encryption methods requires regeneration of the header, which is the same for all methods. Furthermore, only C1 and C2 require exchange, the communication complexity of which is of the same size as the two multiplicative elements. The proxy notification of deletion is similar to adding a friend by an ID with the size of $\log_2 n$ (e.g. adding a friend), which is rather trivial and ignored.

To access the data in ASBE, CP2, and N1 methods, we only need to receive the generated header and N2 method, which requires receiving the public key to which the 2n + 2 value is added. Eventually, the communication complexity becomes 2n + 4. In CP-ABE and EASiER methods, it is necessary to investigate the attributes and receive the parameters pertaining to CT, equal to 2y + 2.

Considering all the above-mentioned cases and the advantages of the methods presented here compared to other methods, if the communication complexity is considered, the N1 method is recommended. The N1 method outperformed all other methods except CP2. However, its performance was quite similar to that of CP2 and had only more overhead in the setup section. Such an overhead value is negligible due to the advantages of the presented method. In a probable scenario, if there are numerous friends with a shared attribute, and all of them intend to communicate with each other, or multiple users in the same group simultaneously intend to send messages to one another, then there is a need to receive a private key and a public key different for each friend owing to one to multiple CP2 in the encryption methods, where the keys are to be defined and stored. Here, the size of the public key alone equals 2n + 2; therefore, both the communication complexity and the storage complexity are significantly increased. In this case, because both N1 and N2 are many-to-many and use the same public key for all friends, they outperform CP2. Thus, the N1 communication overhead in this scenario is less than that of all methods. The use of a proxy in the proposed method has caused a part of the user's communication overhead to be decreased and deposited. Here, the proxy takes over some of the communication tasks between the user and CA. Figures 9, 10, 11, 12 and 13 compare N1 and N2 methods to others.

5.1.2 Computation complexity in the single-layer system

In addition to using asymmetric encryption, EASiER, CP-ABE, and CP2 methods employ symmetric encryption, which is ignored in this part of computation complexity. The most

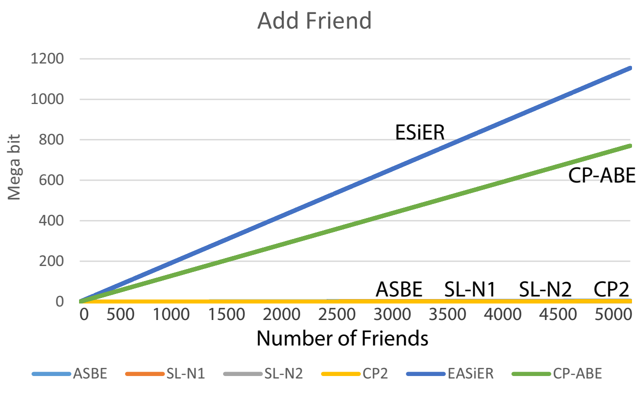

Fig. 9 Communication complexity for adding friends

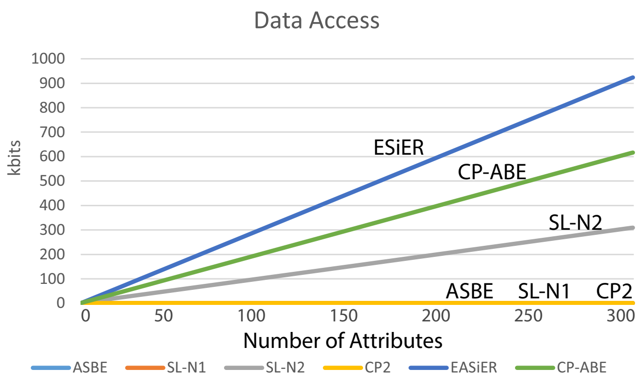

Fig. 10 Communication complexity for accessing data

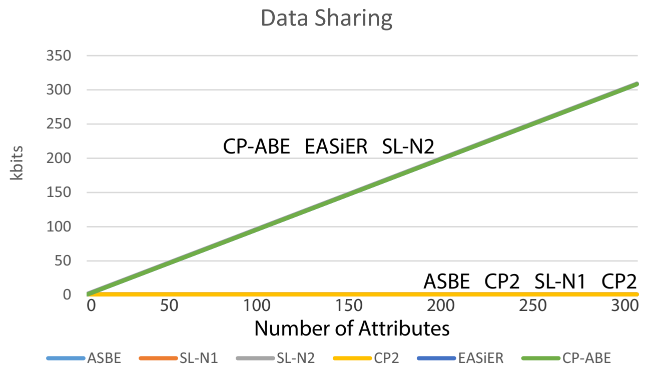

Fig. 11 Communication complexity for sharing data

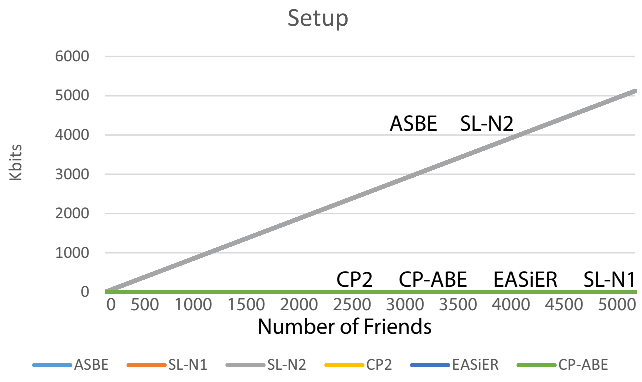

Fig. 13 Communication complexity for setup

costly operations in encryption based on a bilinear map, particularly broadcast encryption, are the computation of point pairing, point multiplication, and point exponentiation. Table 2. presents these computations in categories.

It should be noted that Table 2 indicates only the complex computations of the user side. Other computations are ignored, since they occur in the areas of social network and its collaborating parties with high computing power. User-side computation is important, since mobile social networks are the bottleneck of smartphones, because they usually have low computing power. Moreover, our proposed method is based on the fact that the end-user is involved in encryption and decryption; therefore, the computation complexity is significant and to be examined particularly on the user side. As Table 2 showed, the key preparation and generation computations are identical in ASBE, N1 and N2 methods based on broadcast encryption. A pairing is required to generate $e(g,g)^{\alpha^{2n+1}}$. This value can be received from $e(g_{2n+1-i}, g_i)$, performed by a proxy or the key operator himself to reduce the pairing load for the user by default. To generate a public key containing 2n elements, there is a need for 2n exponentiation operations. In generating the key to generate $d_{i.\ j}$, $T_{i.\ j}$ two exponentiation operations are required; as mentioned before, they are the responsibility of the key operator in all methods. However, in EASiER and CP2 methods, the user side has to perform computation operations, resulting in a computation overhead. CP-ABE is also employed to prepare the operator key, where the amount of computation on the user side is zero. In all four methods of ASBE, CP2, N1, and N2, encryption is needed to exponentiate three elements, two elements of the multiplicative group of origin, and one element of the target

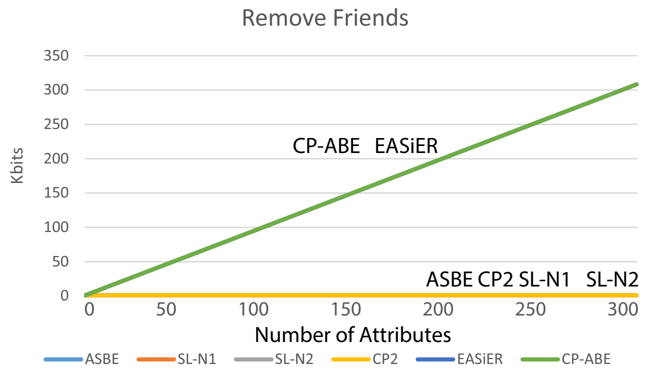

Fig. 12 Communication complexity for removing friends

Table 2 Computation complexity

Method	Opration	Pairing	EXP	MUL
EASiER	Setup*	1	2	0
	Keygen*	0	2 N+1	N/2
	Encrypt*	0	2y+2	0
	Revoke*	0	0	0
	Convert	0	y	0
	Decrypt*	3y	y	2y
CP2	Setup*	1	2n	0
	Keygen*	0	1	0
	Encrypt*	0	3	0
	Decrypt*	2	2	1
ASBE	Setup	1	2n+1	0
	Keygen	0	2	0
	Encrypt*	0	3	n
	Decrypt*	2	2	1
CP_ ABE	Setup	1	2	0
	keyGen	0	3 N/2+1	N/2
	Encrypt*	0	2y+2	0
	Delegate	0	3 N/2	3 N/2
	Decrypt *	2y	0	0
SL-N1	Setup	1	2n+1	0
	Keygen	0	2	0
	Encrypt*	0	3	0
	Decrypt*	1	0	0
SL-N2	Setup	1	2n+1	0
	Keygen	0	2	0
	Encrypt*	0	3	0
	Decrypt*	1	2	1

Computations on behalf of the user are shown with *

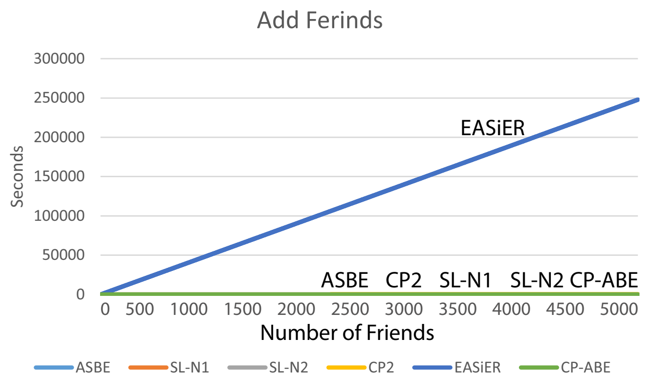

Fig. 14 Computation complexity for adding friends

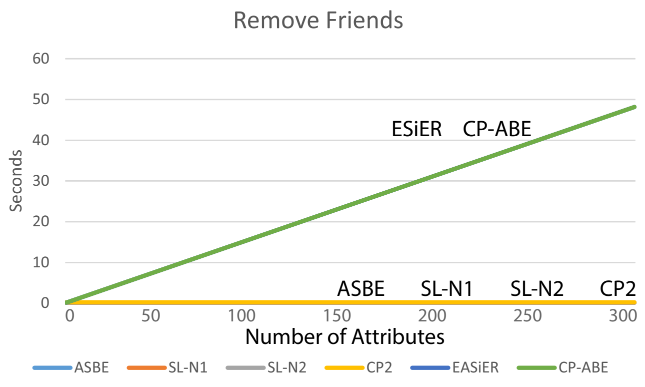

Fig. 16 Computation complexity for removing friends

group. Encryption in EASiER and CP-ABE methods requires 2y + 2 exponentiation operations depending on the number of attributes. Decrypting the ASBE method requires two pairing operations, two exponentiation operations and one multiplication operation. Owing to outsourcing, the pairing operation is reduced to one operation in both N1 and N2 methods. In addition, in the N1 method, no further computation is required on the user side. Furthermore, in the N2 method, in addition to a pairing operation, there is a need for one exponentiation operation and one multiplicative operation. Both methods are required to use the public key to generate $g^{p_i(\alpha)}, g^{r_i}$ $h_i(\alpha)$, and ultimately a multiplication operation, since the operation is conducted in the first part of the map.; In the N1 method, this operation is generally outsourced; however, in the N2 method, it is calculated by the user. The total of user-side computations is mentioned for both foregoing methods, and in the CP2 method, there is a need for two pairing operations, two exponentiation operations and one multiplication operation for decrypting operations. The CP-ABE method requires 2y pairing operations depending on the number of attributes. Moreover, the EASiER method requires 3y pairing, y exponentiation, and 2y multiplicative operations.

As seen above, both N1 and N2 methods have a lower computation complexity compared to all methods. Now, if the communication complexity part of the scenario is changed, in the same manner as the many-to-many communication is established among many users, the optimality of computation complexity will be more observable in both N1 and N2 methods As mentioned, the proposed method decreases the computational overhead imposed on the user by outsourcing part of the computations. In addition, owing to the nature of its broadcast, it eliminates the need for separate encryption for each user and protects privacy and security, which is a significant advantage over other methods in large groups of users. Figures 14, 15, 16, 17 and 18 compare the computation time by the responsibility of the user in both N1 and N2 methods with other methods.

5.1.3 Storage complexity of the single-layer system

In the usual ASBE method, two decryption and encryption keys ($d_{i.\ j}$, $T_{i.\ j}$) are required for each n user, which eventually equal 2n; a public key is further required in decrypting operations. This key is made up of 2n + 2 multiplicative elements that need to be stored. Thus, the total keys to be stored in this method are 4 N + 2. In the N1 method, owing to the assignment of the first pairing to the proxy, these keys are assigned to the proxy for storage, and there is only a need for n private keys per person for the end-user decryption stored in the end-device. Moreover, it requires the storage of two multiplicative elements to compute $e(g, g)$, In N2 method, this value is reduced to one element, r_i, in addition to two multiplicative

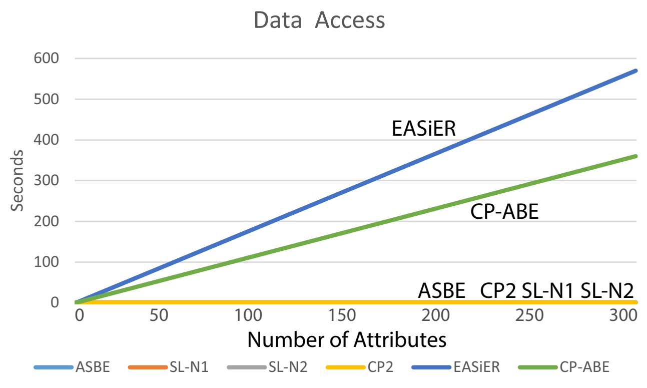

Fig. 15 Computation complexity for accessing data

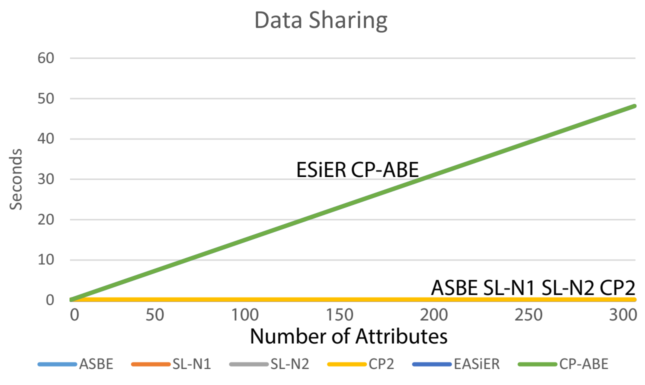

Fig. 17 Computation complexity for sharing data

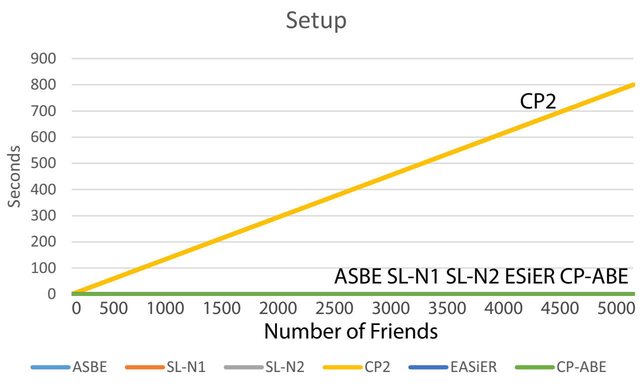

Fig. 18 Computation complexity for setup

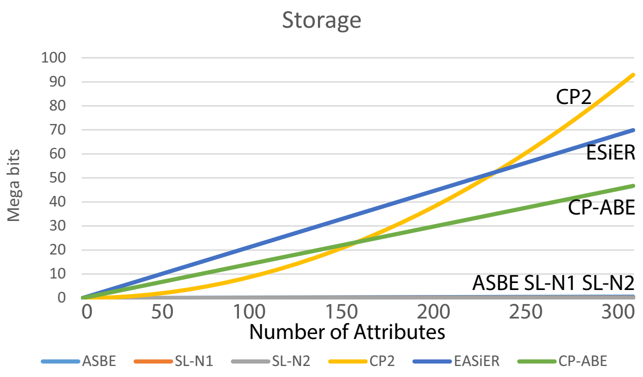

Fig. 19 Storage complexity

elements such as N1 require storage where the overhead computation of such method will reach value 3. Note that the N2 method still requires a public key, which is not always stored with the encrypted text; it is therefore calculated as a communication overhead and ignored here. Storage in the CP2 method, (n + 1) (2n + 2) + n, is much higher than both of our presented methods. In EASiER and CP-ABE, the amount of communication complexity such as CP2 method is high, equaling 4 (n + 1) + n (3 N / 2 + 1) +1 and 3 (n + 1) + n (n + 1), respectively. Here, similar to the two previous scenarios, if the users begin interaction as many-to-many, then the need to store encryption and decryption keys by users, particularly in CP2, is significantly increased. Storing the key elated to proxy in the proposed method decreases the storage overhead. This feature will show itself more in the public key storage. The proposed method contains a much lower overhead than other compared methods do due to the lack of key storage regarding the number of group members and user friends. Table 3 and Fig. 19 compare N1 and N2 methods to the other methods in terms of the storage complexity rate.

5.2 Sub-scenario

To better identify the difference in performance among the various methods in the three selected parameters, a sub-scenario is considered in addition to the above general scenario. In this sub-scenario, the user can have 150 friends, join 50 groups and use 50 features. Moreover, to better fathom the performance summation in the three parameters, we Normalized (by dividing by the maximum) on the sub-scenario data and analyzed the results.

Table 4 shows the different values of adding friend, data sharing, removing friends, accessing data, and setting up in investigating the communication cost in Kb. As observed, the cost in the N1 method is equal to CP2 in all parameters except setup. These two methods have the least communication cost among all the methods. The communication cost of the N2 method is also acceptable, having the least value following the two methods above. In the final line of this Table, assuming that a user uses all operations in the sub-scenario only once, how much will be the communication cost in each method. Here, N1 and cp2 methods have the least value. However, as mentioned before, CP2 is unable in terms of many-to-many communication and in the case of supporting this communication; the communication cost is dramatically increased.

Table 5 shows the computation costs for the sub-scenario, which are presented in terms of computation time in seconds. As observed, both N1 and N2 methods in the single-layer system have the least computation time compared to all other methods, enjoying convincing performance. The computation time in the ASBE method is close to the one in the proposed methods; however, their use in mobile social networks is not justified due to the defects and disadvantages of the system

Table 3 Storage complexity

Method	Storage
EASiER	4(n+1)+n(3 N/2+1)+1
CP2	(n+1)(2n+2)+n
ASBE	(4n+2)
CP_ABE	3(n+1)+n(N+1)
SL-N1	n+2
SL-N2	3

Table 4 Communication complexity in the sub- scenario

	ASBE	SL-N1	SL-N2	CP2	EASiER	CP-ABE
Add Friends	150	75	0	75	33,825	22,575
Data sharing	1	1	51	1	50	50
Remove Friends	1	1	1	1	50	50
Data Access	1	1	51	1	149	100
Setup	150	1	150	0	0	1.5
Sum	303	79	253	78	34,074	22,776.5

Table 5 Computation complexity in the sub- scenario

	ASBE	CP2	EASiER	CP-ABE	SL-N1	SL-N2
Add Friends	0	12	7437	0	0	0
Data sharing	0.16	0.24	8	8	0.16	0.16
Remove Friends	0.16	0.24	8	8	0.16	0.16
Data Access	1.37	1.37	93.1	58.8	0.6	0.77
Setup	0	24.6	0.76	0	0	0
Sum	1.69	38.45	7546.86	74.8	0.92	1.09

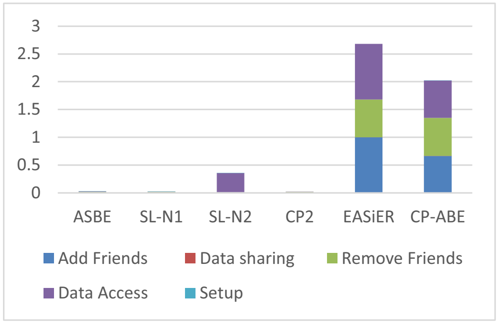

Fig. 20 Normalizing results of the communication complexity

Table 6 Storage complexity in the sub-scenario

	ASBE	CP2	EASiER	CP-ABE	SL-N1	SL-N2
Storage	99	2524.5	11,150	7449.5	25.5	1.5

Table 8 Normalizing results of the computation complexity

	ASBE	CP2	EASiER	CP-ABE	SL-N1	SL-N2
Add Friends	0	0.0016	1	0	0	0
Data sharing	0.0136	0.0204	0.6803	0.6803	0.0136	0.0136
Remove Friends	0.0136	0.0204	0.6803	0.6803	0.0136	0.0136
Data Access	0.0147	0.0147	1	0.6316	0.0064	0.0083
Setup	0	1	0.0309	0	0	0

similar to those offered in our framework and even other similar methods. The last line of this table further indicates the total time of operations by the user, assuming that each evaluated operation is performed only once.

Table 6 presents the storage complexity values in the sub-scenario for all the evaluated methods in Kb. As shown, the N2 method has the least amount of such complexity among all methods in the single-layer system in both N1 and N2.

5.3 Normalization of results

This section summarizes the performance of the proposed methods over other methods through normalization by dividing it by the maximum, where all values are located between zero and one, and their analysis is better. Here, the same sub-scenario is assumed. To obtain the values, each user has 150 friends with 50 attributes or memberships in 50 groups.

Table 7 and Figure 20 show the communication complexity based on adding friends, data sharing, removing friends, accessing data and setting up in the social network to compare the performance of different methods to that of the methods presented by using normalization. This figure shows that both N1 and N2 methods in the single-layer system performed well in all operations and generally performed better than other methods did.

Table 8 and Fig. 21 indicate the computation complexity of all operations in all methods. It is clearly understood that in the single-layer system, both N1 and N2 methods outperformed all methods in all operations.

Table 9 and Fig. 22 present the storage complexity, showing that N1 and N2 methods in the single-layer system had the optimal conditions.

5.4 Comparison of the proposed framework to others

In this section, the proposed method is compared to others in terms of protecting users' privacy against social network

Table 7 Normalizing results of storage complexity

	ASBE	CP2	EASiER	CP-ABE	SL-N1	SL-N2
Add Friends	0.004435	0.002217	1	0.667406	0.002217	0
Data sharing	0.000008	0.000008	0.000408	0.000408	0.000008	0.000416
Remove Friends	0.013605	0.013605	0.680272	0.680272	0.013605	0.013605
Data Access	0.006711	0.006711	1	0.671141	0.006711	0.342282
Setup	0.006666	0	0	0.000066	0.000044	0.006666

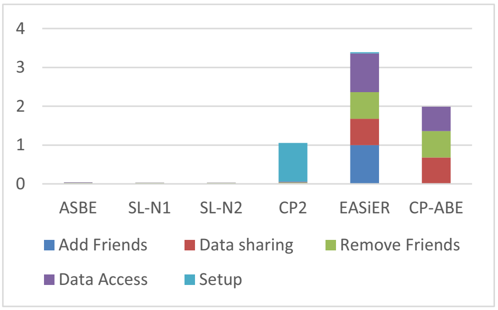

Fig. 21 Normalizing results of the computation complexity

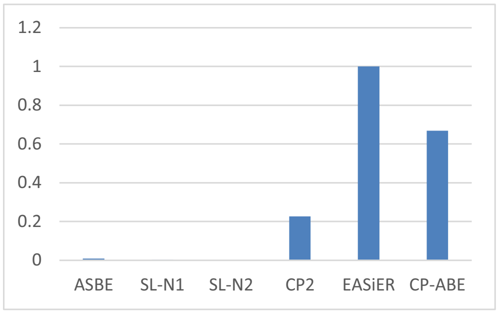

Fig. 22 Normalizing results of storage complexity

providers and unauthorized users, protecting the users' privacy relation, supporting many-to-many encryption, utilizing proxy to reduce the computational load, protecting the device, supporting sending and receiving all types of messages safely, and providing suitable access control. Table 9 lists the results.

The comparison results in Table 9 reveal that the proposed method is a comprehensive and safe design with high privacy and practicability.

6 Conclusion

This paper presented a broadcast encryption using attribute-based encryption properties that optimally enable the use of mobile social networks for privacy protection. By outsourcing the costs of computation and maintaining broadcast encryption keys, this approach was made applicable in the real world. It provides the user with complete privacy control in the social network structure. In this framework, owing to the separation of the components, the proposed social network takes control over data from social network providers and presents them to users, allowing users to assign any number of attributes and any order to their friends or created groups. Data sharing and access permissions for users are provided in completely secure terms. The distribution of keys in this framework is in a way ending up in the end-user device, and there is no need to be concerned about their compromised safety. Unauthorized users cannot be accessed due to encryption and data storage. Identification of users, friends, and the relationships between them in this framework is secure, so that their privacy is not violated. To survey our proposed framework, we investigated the parameters of communicational and computational complexity in the basic operations of social networks, including removing and adding friends, sharing and accessing data, and setting up. Moreover, we examined the complexity of storage. The investigated operations are among the most important challenges of social networks. The investigations indicate that in our method, the communicational complexity in all operations (except Setup and only compared to the CP2 method, which is also slightly different) as well as in the computational complexity, contains the lowest value and as a result, the lowest communicational and computational overhead than the

Table 9 Results of comparison of the proposed framework with others

Methods		EASiER	CP-ABE	ASBE	CP2	SL-N1	SL-N2
Protection against breach of confidentiality by OSN providers		✓	✗	✗	✓	✓	✓
Protection against breach of confidentiality by non-friends		✓	✓	✓	✓	✓	✓
Relational confidentiality		✗	✗	✗	✓	✓	✓
Many to many encryption		✗	✗	✗	✗	✓	✓
Use of proxy for reducing computation load		✗	✗	✗	✗	✓	✓
Protection on device		✓	+	✓	✓	✓	✓
Multimedia & document		✓	Not menti-oned	✓	✓	✓	✓
Access control	Fine grained	✓	✓	+	+	✓	✓
	Flexible	+	✓	✓	✓	✓	✓
	Dynamic	+	+	+	+	✓	✓

Comparison of the privacy methods based on the OSN privacy requirements, assuming full satisfaction of the requirement by ✓, partial satisfaction of the requirement by +, and failing to satisfy the requirement by ✗

compared method. Furthermore, the storage complexity and the storage overhead are significantly lower than those of the compared methods. Our proposed framework provides all the privacy requirements for mobile social network users with the least cost.

For future work, it is recommended that: (1) proxy and cloud components be combined in this framework; (2) For each feature that the user specifies for group members or those who have access to shared data, a key should be defined to combine them to apply the desired access control with more freedom; (3) CA be removed and its tasks be left to users, resulting in the decentralized performance of these tasks. According to the obtained experiences, it is recommended that the proposed framework be implemented in safety encryption methods having heavy computational and key agreement protocols. Moreover, the use of broadcast encryption methods in servers and file sharing and transfer networks seems appropriate. Meanwhile, the use of this framework can be appropriate for encryption in social networks based on video and audio. Applying this method in the satellite global positioning system can also be effective to protect users' privacy and security in military and commercial uses.

References

1. Malekhosseini R, Hosseinzadeh M, Navi K (2018) An investigation into the requirements of privacy in social networks and factors contributing to users ' concerns about violation of their privacy. Soc Netw Anal Min
2. Cockcroft S, Clutterbuck P (2001) Attitudes towards information privacy, in Australasian Conference on Information Systems, School of Multimedia and Information Technology, Southern Cross University,pp. 1–11
3. Gross R, Acquisti A (2005) Information revelation and privacy in online social networks, in Proceedings of the 5nd ACM workshop on Privacy in the Electronic Society,pp.71–80, USA
4. Chang W, Wu J, Tan CC (2011) Friendship–based location privacy in mobile social networks. International Journal of Security and Networks, pp.226–236
5. De Cristofaro E, Soriente C, Tsudik G, Williams A (2012) Hummingbird: privacy at the time of twitter, in 2012 IEEE Symposium on Security and Privacy, pp. 285–299
6. Krishnamurthy B, Wills CE (2009) On the leakage of personally identifiable information via online social networks," Proceedings of the 2nd ACM workshop on Online social networks. ACM, pp.7–12
7. Jahid S (2013) Social networking: security, privacy, and application. University of Illinois at Urbana-Champaign
8. Chen S, Williams M-A (2009) Privacy in social networks: A comparative study. PACIS 2009 Proceedings
9. Oukemeni S, Rifà-pous H, Manuel J, Puig M (2019) Privacy analysis on microblogging online social networks : a survey. ACM Computing Surveys (CSUR), 52(3)
10. Ying X, Pan K, Wu X, Guo L (2009) Comparisons of randomization and K-degree anonymization schemes for privacy preserving social network publishing, in Proceedings of the 3rd Workshop on Social Network Mining and Analysis - SNA-KDD '09, pp. 1–10
11. Liu K, Terzi E (2008) Towards identity anonymization on graphs. Proceedings of the 2008 ACM SIGMOD international conference on Management of data - SIGMOD 08:93–106
12. Hay M, Miklau G, Jensen D, Towsley D, Weis P (2008) Resisting structural re-identification in anonymized social networks. Proc VLDB Endow, Aug 2008 1(1):102–114
13. Boldi P, Bonchi F, Gionis A, Tassa T (2012) Injecting uncertainty in graphs for identity obfuscation. Proc VLDB Endow, Jul 2012 5(11):1376–1387
14. Baden R et al (2009) Persona: an online social network with user-defined privacy. in Proceedings of the ACM SIGCOMM 2009 conference on Data communication - SIGCOMM '09 39(4):135–146
15. Jahid S, Mittal P, Borisov N (2011) EASiER: Encryption-based access control in social networks with efficient revocation. Proc. 6th Int. Symp. Information, Comput. Commun. Secur. ASIACCS 2011:411–415
16. Tootoonchian A, Gollu KK, Saroiu S, Ganjali Y, Wolman A (2008) Lockr," in Proceedings of the first workshop on Online social networks - WOSP '08, pp.43–48
17. Lucas MM, Borisov N (2008) FlyByNight," in Proceedings of the 7th ACM workshop on Privacy in the electronic society - WPES '08, pp.1–8
18. Luo W, Xie Q, Hengartner U (2009) FaceCloak: an architecture for user privacy on social networking sites, in 2009 International Conference on Computational Science and Engineering, pp. 26–33
19. Guha S, Tang K, Francis P (2008) "NOYB," in Proceedings of the first workshop on Online social networks - WOSP '08,pp.49–54
20. Raji F, Miri A, Jazi MD (2013) CP2: cryptographic privacy protection framework for online social networks. Comput Electr Eng 39(7):2282–2298
21. Xiao X, Chen C, Sangaiah AK, Hu G, Ye R, Jiang Y (2018) CenLocShare: A centralized privacy-preserving location-sharing system for mobile online social networks. Futur Gener Comput Syst 86:863–872
22. Van Eecke P, Truyens M (2010) Privacy and social networks. Comput Law Secur Rev 26(5):535–546
23. Chin A, Zhang D (2013) Mobile social networking: an innovative approach. Springer Science & Business Media
24. Ardagna C, Jajodia S, Samarati P, Stavrou A (2013) Providing users' anonymity in mobile hybrid networks. ACM Transactions on Internet Technology (TOIT) 12:1–33
25. Fu S, He L, Liao X, Huang C (2016) Developing the cloud-integrated data replication framework in decentralized online social networks. J Comput Syst Sci 82(1):113–129
26. He K, Weng J, Liu JN, Liu JK, Liu W, Deng RH (2016) Anonymous identity-based broadcast encryption with chosen-ciphertext security, ASIA CCS 2016 - Proc. 11th ACM Asia Conf. Comput. Commun. Secur.,pp. 247–255
27. Wang G, Feng J, Bhuiyan MZA, Lu R (2019) Security, privacy, and anonymity in computation, communication, and storage. 12th International Conference, SpaCCS 2019, Atlanta, GA, Springer International Publishing, vol. 11611, USA
28. Abdulla AK, Bakiras S (2019) HITC : Data Privacy in Online Social Networks with Fine-Grained Access Control, In Proceedings of the 24th ACM Symposium on Access Control Models and Technologies, no. i, pp. 123–134 (2019)
29. Shamir A, Adi (1979) How to share a secret. Commun ACM 22(11):612–613
30. Shamir A (1984) Identity-based cryptosystems and signature schemes, in Advances in Cryptology, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 47–53
31. Sahai A, Waters B (2005) Fuzzy identity-based encryption. Springer, Berlin, Heidelberg:457–473
32. Ostrovsky R, Sahai A, Waters B (2007) Attribute-based encryption with non-monotonic access structures. In Proceedings of the 14th

ACM conference on Computer and communications security , pp. 195–203
33. Han J, Susilo W, Mu Y, Yan J (2012) Privacy-preserving decentralized key-policy attribute-based encryption. IEEE Transactions on Parallel and Distributed Systems 23:2150–2162
34. Goyal V, Pandey O, Sahai A, Waters B (2006) Attribute-based encryption for fine-grained access control of encrypted data," in Proceedings of the 13th ACM conference on Computer and communications security - CCS '06, pp.89–98
35. Bethencourt J, Sahai A, Waters B (2007) Ciphertext-policy attribute-based encryption," in 2007 IEEE Symposium on Security and Privacy (SP '07), pp. 321–334
36. Sun X, Yao Y, Xia Y, Liu X, Chen J, Wang Z (2016) Towards efficient sharing of encrypted data in cloud-based mobile social network.," KSII Transactions on Internet & Information Systems, 10.4
37. Fiat A, Naor M (1993) "Broadcast Encryption," in Advances in Cryptology — CRYPTO' 93, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 480–491
38. Boneh D, Mironov I, Shoup V (2003) A secure signature scheme from bilinear maps, In: Cryptographers' Track at the RSA Conference. Springer, Berlin, Heidelberg, pp. 98–110
39. Boneh D, Franklin M (2001) international, "Identity-based encryption from the Weil pairing, In: Annual international cryptology conference. Springer, Berlin, Heidelberg, pp. 213–229
40. Boneh D, Gentry C, Waters B (2005) Collusion resistant broadcast encryption with short ciphertexts and private keys," In Annual International Cryptology Conference , no. 1, pp. 258–275
41. Malek B, Miri A (2012) Adaptively secure broadcast encryption with short ciphertexts. Int J Netw Secur 14(2):71–79
42. Mahmoodzadeh KS, Safi SM (2018) Survey on common broadcast encryptions. Int J Comput Sci Netw Solut 6(3):1–7
43. Bethencourt J, Sahai A, Waters B (2007) Ciphertext-policy attribute-based encryption, In: 2007 IEEE symposium on security and privacy (SP'07). IEEE, pp. 321–334
44. Gao C, Cheng Q, Li X (2018) Cloud-assisted privacy-preserving profile-matching scheme under multiple keys in mobile social network. Cluster Comput 22(1):1655–1663
45. Ali S, Rauf A, Islam N, Farman H (2017) A framework for secure and privacy protected collaborative contents sharing using public OSN. Cluster Comput. 22(3):7275–7286
46. Facebook Help (2018) [Online]. Available: https://www.facebook.com/help/%0D%0A