A General Framework for Group Authentication and Key Exchange Protocols

Huihui Yang(✉), Lei Jiao, and Vladimir A. Oleshchuk

Department of Information and Communication Technology, University of Agder, Kristiansand, Norway
{huihui.yang,LeiJiao,vladimir.oleshchuk}@uia.no

Abstract. In this paper, we propose a novel framework for group authentication and key exchange protocols. There are three main advantages of our framework. First, it is a general one, where different cryptographic primitives can be used for different applications. Second, it works in a one-to-multiple mode, where a party can authenticate several parties mutually. Last, it can provide several security features, such as protection against passive adversaries and impersonate attacks, implicit key authentication, forward and backward security. There are two types of protocols in our framework. The main difference between them is that the authenticator in Type II has a certificate while in Type I does not. Under the general framework, we also give the details of protocols based on Diffie-Hellman key exchange system, and discrete logarithm problem (DLP) or elliptic curve discrete logarithm problem (ECDLP) based ElGamal encryption respectively. Session keys will be established at the end of each session and they can be utilized later to protect messages transmitted on the communication channel.

Keywords: Group authentication · Diffie-Hellman key exchange · Discrete logarithm problem · Elliptic curve discrete logarithm problem

1 Introduction

Online social networks (OSNs) are platforms offering social services in an online format, and they became very popular during recent years. OSNs can be used in many ways, such as news sharing, group chatting and video conferences and so on. For all these applications, authentication is of great importance. Consider the example of group chatting. All group members should be whom one expects to communicate with, and messages delivered between them should also be protected. It is also very important for servers to authenticate their clients. However, due to the large number of clients for most OSNs, the time spent by servers to authenticate clients may become a bottleneck of the whole service if it is done in the traditional one-to-one mode. Therefore, an efficient and secure protocol for authenticating and key exchanges is needed.

J.-L. Danger et al. (Eds.): FPS 2013, LNCS 8352, pp. 31–45, 2014.
DOI: 10.1007/978-3-319-05302-8_3, © Springer International Publishing Switzerland 2014

Authentication can be achieved by usernames and passwords [1,2] or public certificates [3]. Password based approach is usually selected for the authentication in a client and server mode, where usernames and passwords are required. For certificate based authentication, however, a public key infrastructure is needed to be built first and then an initiation phase for key distribution is required. Besides, key based authentication works in a one-to-one mode, so only one user or client can be authenticated each time. To save both time and bandwidth, group authentication [4] is proposed in this paper. Under most circumstances, group authentication is related to group key management in multimedia or wireless network and etc. It is mainly used to prove that a member belongs to a certain group without revealing its identity. However, we give a new definition of group authentication in this paper, where all members in a group can be authenticated at one time. The main difference between the traditional definition and ours is that the former aims at authentication without revealing anonymity while ours is to save time and increase authentication efficiency. This new definition of group authentication has already been proposed in [2], which is password based authentication and can be used to verify multiple users' identities at the same time. We will adopt this new definition in our paper.

Compared with existing work, our paper mainly has three contributions. First of all, the authentication in all our protocols is group based, and thus it is more effective and can save both bandwidth and time. Secondly, our framework is a general one and can be based on different cryptographic primitives. Meanwhile, we give two detailed applications using DLP [5] and ECDLP [6] based ElGamal encryption [7]. Finally, our protocols can satisfy security requirements such as mutual and implicit key authentication, protection against passive adversaries and impersonation attacks and forward and backward secrecy. Later, we will give detailed proofs for each of these security requirements.

2 Related Work

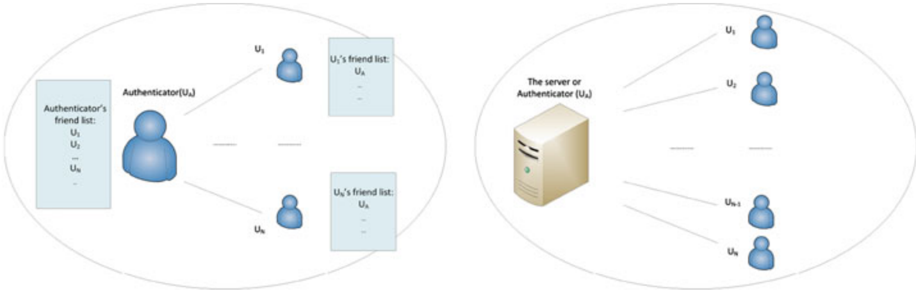
A lot of research has already been done in the area of authentication and key exchanges [8,9]. We will mainly discuss those related to group authentication rather than authentication in the one-to-one mode. As mentioned in Sect. 1, the traditional group authentication is to prove that a member belongs to a certain group. Among all approaches to achieve it, group signatures [10] and ring signatures [11] are often used. In group signatures, a member of a group or the signer generates a signature, a verifier can check whether the signer belongs to this group or not, without knowing the singer's identity. However, the manager of this group can "open" the signature and thus the identity of the signer will be revealed if necessary. Ring signature is based on public key system. There is no group manager, and a signer can choose anyone whose public key is registered in a trusted authority to create a random group. The signer uses its own secret key and other members' public keys to generate a signature. The other members can deny that the signature is created by them and the real identity of the signer can be revealed in this way.

There are also some other researches based on group or ring signatures. The protocol in [12] is based on digital signature algorithm (DSA) [13] and is designed to authenticate vehicles. However, the sender or authenticator deals with the received responses from each vehicle separately because each vehicle encrypts its response by the authenticator's public key. As a result, this protocol does not contribute much to saving computational cost. In [14], the protocol is DSA and DLP based. It can be used to authenticate a group member, subgroups or all members in a group, but it differs from our protocols in several ways. First of all, in their protocol there should be a group leader or a trusted party, who signs the message first and checks whether other users' signatures are valid or not. Secondly, it is designed for an outsider to authenticate a group who shares a public key, while the outsider does not need to know any of the group members. Similarly, the model proposed in [15] is also designed for an outsider to verify the signature of a group, so theoretically it can be used to authenticate a group of members. However, the group should be stable and every member should have a certificate. In [16], a series of protocols are designed for batch or group authentication in a one-to-multiple mode. They can be applied to a scenario where strangers are to be authenticated by a party under the help of a trusted friend in a P2P based decentralized social network. They propose protocols based on one-way hash function [17], ElGamal signature [7,18] and certificates respectively. However, even though their one-way hash function based protocol in [16] can be utilized to authenticate a group of members, the computational cost does not decrease compared with that in a one-to-one mode, so it does not benefit much in time saving on the authenticator's side. Their work differs from ours in several ways. The most significant difference is that we emphasis on a general framework where several cryptographic primitives can be used to it, rather than specific primitives. Besides, the protocols in [14,15] do not provide key exchange functionality, and they are not fit for the authentication inside a group. In this paper, we propose a framework for the mutual authentication within a group and it also provides key exchanges.

The rest of the paper is organized as follows. In Sect. 3, two usage scenarios are introduced first. Then the main notations and parameters that will be used later in our framework are explained and the framework for both types of protocols is described. Next, two examples are given to demonstrate how our framework works. Finally, a formal description of what cryptographic primitives can be used to our framework is presented. The correctness and security analysis are presented in Sects. 4, 5 respectively. In Sect. 6, we will give some comparisons of the computation and communication costs of our protocols and the traditional one-to-one mode ones. Finally, we conclude this paper in the last section and give some suggestions about how to apply our framework.

3 The General Framework

In this section, we firstly explain two typical usage scenarios where our framework can be applied, and then the general parameters, notations, message flows of our



(a) Relations between the Authenticator and his or her Friends

(b) Relations between the Server and its Clients

Fig. 1. Relations of two parties in two scenarios. **a** Relations between the authenticator and his or her friends. **b** Relations between the server and its clients

general framework are presented. Next, we implement some specific primitives of our framework to demonstrate how it works. Finally, we describe which kinds of cryptographic primitives can be applied to our framework.

3.1 Two Usage Scenarios

All protocols in our framework are suitable for two scenarios, illustrated in Fig. 1. In scenario 1(a), an ordinary user temporarily creates a group with his or her friends and authenticates each of them. Scenario 1(b) is for a server to authenticate several of its clients. Both the group initiator and the server shares some secrets with his or her friends or its clients, and it will use these secrets for mutual authentication later. Our protocols have two goals: mutual authentication and session key agreement. At the end of each session, session keys are established. The group session key is generated by the authenticator and distributed to each group member, while the other session keys are computed according to Diffie-Hellman [19] key exchange. In scenario 1(b), however, only session keys between the server and its clients are established. Our protocols can be divided into two types. The main difference between them is that the authenticator in Type I does not have a certificate while a certification is needed in Type II, and thus the authenticator is authenticated differently in protocols of Type I and Type II.

3.2 Parameters and Notations

The parameters and notations for all protocols are listed in Table 1. Among them, we will only give some explanations to k_i. For both scenarios mentioned in Sect. 3.1, k_i is a long time shared secret, generated by U_A and has been delivered to U_i via a safe channel in advance. For the first scenario, since we assume that the number of U_A's friends is not big, it is practical to generate enough pairwise

Table 1. Parameters and notations

Symbol	Description
U_A	The authenticator to authenticate a group of users or clients denoted by $\mathbb{U}$
$\mathbb{U}$	A user group to be authenticated, $\mathbb{U} = \{U_1, U_2, \ldots, U_N\}$
N	The number of members in group $\mathbb{U}$
ID_A	The identity of U_A
ID_i	The identity of U_i, $i \in \{1, \ldots, N\}$
UID	Identities of all members in $\mathbb{U}$, $UID = \{ID_1, ID_2, \ldots, ID_N\}$
$H()$	One-way hash function with the output of length l, $H() : \{0,1\}^* \rightarrow \{0,1\}^l$
MAC	The message authentication code generated by a keyed hash function
ξ	$\xi = \{ID_A, UID\}$, message used for group authentication
K_G	Group session key, generated by U_A during a specified session, one time use
SK_{Ai}	Session key between U_A and U_i in a specified session, one time use
SK_{ij}	Session key between U_i and U_j in a specified session, one time use
SK_A	U_A's private key
$SIGN_K(m)$	The signature of message m with private key K
KP_A	Key parameters generated by U_A, $KP_A = \{g^{m_1}, \ldots, g^{m_N}\}$ in DLP based protocols and $KP_A = \{m_1 G, \ldots, m_N G\}$ in ECDLP based protocols
KP_U	Key parameters generated by members in $\mathbb{U}$, $KP_U = \{g^{n_1}, \ldots, g^{n_N}\}$ in the DLP based protocol and $KP_U = \{n_1 G, \ldots, n_N G\}$ in the ECDLP based protocol
k_i	$k_i \in \mathbb{K} = \{k_1, k_2, \ldots, k_N\}$. It is a long time shared secret between U_A and U_i. k_i, k_j $(1 \leq i, j \leq N, i \neq j)$ are pairwise prime
t_i	One-time nonce, used to make sure of the freshness of a session
q, p	Large prime numbers, and $q = 2p + 1$
g	The generator of the cyclic multiplicative group G_q
y_A	A secret shared between U_A and members in $\mathbb{U}$
y_i	$y_i = g^{x_i}/x_i G$ in the DLP and ECDLP based protocols respectively It is a secret shared between U_A and U_i
G	Base point of the selected elliptic curve

prime numbers k_i. In the second scenario, there can be a huge number of users per server. In this case, U_A can generate several groups of different k_i, where they are pairwise prime within the same group. As a result, U_A can authenticate clients with k_i within the same group at one time. However, this mechanism is needed only when the number of clients exceeds the threshold that the server can deal with.

3.3 Message Flows

There are four steps in our framework and the message flows are illustrated in Fig. 2. During the message flows, C_i, V_i and W_i all depends on different

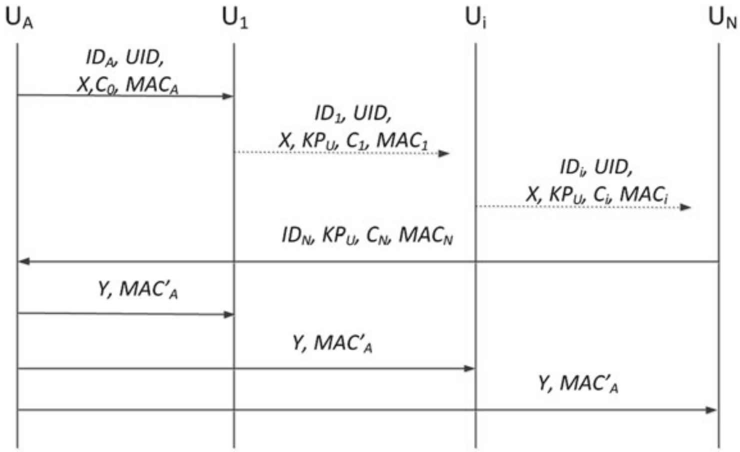


Fig. 2. Message flows of the general framework

cryptographic primitives and will be explained later. The details of the four steps of message flows are as follows:

(1) $U_A \rightarrow U_1 : ID_A, UID, X, C_0, MAC_A$.
(2) $U_i \rightarrow U_{i+1} : ID_i, UID, X, KP_U, C_i, MAC_i$, where $1 \leq i \leq N-1$.
(3) $U_N \rightarrow U_A : ID_N, KP_U, C_N, MAC_N$.
(4) $U_A \rightarrow \mathbb{U} : Y, MAC'_A$.

(1) U_A initiates a new session in this step. It calculates X by formula (1) using the Chinese remainder theorem (CRT) [20], but it has to be sure that $V_i \oplus k_i < k_i$ $(1 \leq i \leq N)$ (The details will be explained later in Sect. 3.4.).

$$X \equiv V_i \oplus k_i \pmod{k_i}, \text{where } 1 \leq i \leq N. \tag{1}$$

Here, V_i should contain the identity information of both U_A and U_i, group session key K_G, session key parameters and a cryptographic primitive h_i for U_A's authentication. We use one-way hash functions to compute h_i in this paper for simplicity. Then U_A generates C_0 which will be used by U_i to compute C_i. Finally, U_A computes MAC_A and sends it to U_1. After U_1 receives the message, it gets V_1 by $X(\bmod\, k_i) \oplus k_1$, extracts parameters in it and checks MAC_A. If MAC_A is valid, it continues and the authentication for U_A by U_1 is achieved or else it aborts the session. Next it calculates the session key with U_A.

(2) U_i randomly generates its key parameter, appends it to KP_U and computes C_i. Finally, it calculates MAC_i and then sends the message to the next user. After U_{i+1} receives the message, what it does is the same as U_1 in step (1).

(3) The behavior of U_N is the same as U_1. After U_A receives the message from U_N, it checks the validity of MAC_N first. If it is valid, it computes C'_N and checks whether $C'_N = C_N$ holds. If it does, the group authentication is achieved.

(4) U_A generates Y by formula (2) and $W_i \oplus k_i < k_i$ must hold.

$$Y \equiv W_i \oplus k_i \pmod{k_i}, \text{ where } 1 \leq i \leq N. \tag{2}$$

When U_i receives message as step (4) , it proceeds as follows.

(a) Check MAC'_A to make sure that the message is not tampered.

(b) Get W_i by $Y (\bmod k_i) \oplus k_i$.

(c) Retrieve parameters from W_i. If it contains $\{ID_A, ID_i\}$, U_i is successfully authenticated. Then it calculates session keys with U_j $(1 \leq j \leq N, i \neq j)$ and erase its key parameter.

3.4 DLP Based Protocols

The same as the message flows described in Sect. 3.3, the DLP based protocols include four steps and the following are the details.

(1) Let $C_0 = \xi(r) = \xi(g^{r_A})$, where $r_A \in [1, p-1]$ is randomly generated. V_i is derived from V'_i. In the protocol of Type I, $V'_i = \{y_i \oplus K_G, y_i \oplus t_i, g^{m_i}, h_i\}$, where $h_i = H(ID_A \oplus ID_i \oplus y_A \oplus t_i)$, while in Type II, $V'_i = SIGN_{SK_A}\{ID_A, ID_i, K_G,\ g^{m_i}, t_i\}$. As mentioned in Sect. 3.3, we should make sure that $X_i \oplus k_i < k_i$ and $W_i \oplus k_i < k_i$. Suppose the security parameters of k_i and also the length of V'_i is $sl(k_i)$, then the length of k_i should be $l(k_i) > sl(k_i)$ and the highest $l(k_i) - sl(k_i)$ bits are used for the purpose of CRT. When k_i is generated, the highest $l(k_i) - sl(k_i)$ bits are initiated as 1. The highest $l(k_i) - sl(k_i)$ bits of V_i are also initiated as 1 and the rest bits are the same as V'_i, so we can be sure that $V_i \oplus k_i < k_i$ holds. The same approach will be applied to generate W_i.

(2) In protocols of Type I, the authentication of U_A by U_i is obtained by checking h_i. After U_i gets V_i, it extracts t_i, computes $h'_t = H(ID_A \oplus ID_i \oplus y_A \oplus t_i)$ and checks whether $h'_i = h_i$ holds. If it does, U_A is successfully authenticated. Furthermore, $C_1 = \xi(r^{x_1})$ and $C_i = C_{i-1} \times r^{x_i} = \xi(r^{\sum_{t=1}^{i} x_t})$ $(2 \leq i \leq N)$. The session keys SK_{Ai} and SK_{ij} are calculated as $g^{m_i n_i}$ and $g^{n_i n_j}$ respectively.

(3) U_A authenticates the whole user group by checking whether $C'_N = C_N$ holds, where $C'_N = \xi(\prod_{t=1}^{N} y_t^{r_A})$.

(4) Let $W'_i = \{ID_A, ID_i, KP_i\}$ and $KP_i = KP_U - \{g^{n_i}\}$.

3.5 ECDLP Based Protocols

The same as the DLP based protocols, the authentication of U_A by U_i of ECDLP based protocols also depends on the one-way hash function. Parameters and users' behaviors about ECDLP based protocols are listed as follows.

(1) Let $C_0 = G_r = r_{_A}G$, where $r_{_A} \in [1, n-1]$ is randomly selected. In the protocol of Type I, $V_i' = \{y_i \oplus K_G, m_iG, y_i \oplus t_i, h_i\}$, where $h_i = H(ID_A \oplus ID_i \oplus y_A \oplus t_i)$. For the protocol of Type II, however, $V_i' = SIGN_{SK_A}\{ID_A, ID_i, K_G, m_iG, t_i\}$.
(2) In the protocol of Type I, the authentication of U_A by U_i is also obtained by checking h_i. Furthermore, $C_1 = \xi(x_1G_r)$ and $C_i = C_{i-1} + \xi(x_iG_r) = \xi(r_{_A}\sum_{t=1}^{i} x_tG)$ $(2 \le i \le N-1)$. The session keys SK_{Ai} and SK_{ij} are computed as m_in_iG and n_in_jG $(1 \le i, j \le N, i \ne j)$.
(3) U_A authenticates the whole user group by checking whether $C_N' = C_N$ holds, where $C_N' = \xi(\sum_{t=1}^{N} r_{_A}y_i)$.
(4) Let $W_i' = \{ID_A, ID_i, KP_i\}$ and $KP_i = KP_U - \{n_iG\}$.

3.6 Requirements of Cryptographic Primitives

The DLP and ECDLP based protocols described above are only two specific examples and other cryptographic primitives can also be implemented to our framework. Suppose the underlying cryptographic scheme we use is F, $*$ is the operation that joins the results of U_i and U_{i+1} $(1 \le i \le N-1)$ and $\circ$ is the operation that U_A uses for the shared secrets. Let $f_i = F(\xi, k_i)$, where ξ is the message as illustrated in Table 1, f_i is the result of what U_i calculates and k_i is the secret parameter shared between U_A and U_i. If Eq. (3) holds, then it can be applied to our framework.

$$f_1 * \cdots * f_N = F(\xi, k_1 \circ \cdots \circ k_N) \tag{3}$$

The left side of Eq. (3) means that it needs N times operation of F to authenticate all members in $\mathbb{U}$ in the traditional one-to-one mode. However, the same goal can be achieved by only performing F once and $\circ$ N times, where $\circ$ is supposed to be much more time saving compared with F.

In our DLP and ECDLP based protocols, we use one-way hash functions to authenticate U_A, however, many other cryptographic primitives can be used, such as ElGamal signature, DSA and so on, depending on different user scenarios or devices etc.

4 Correctness Analysis

Since our framework does not include specific cryptographic primitives for mutual authentication, we will only give the correctness analysis to DLP and ECDLP based protocols.

4.1 Correctness of DLP Based Protocols

In the protocol of Type I, the authentication for U_A by U_i is promised by checking whether h_i' is equal to h_i, where h_i' is the hash value calculated by U_i. After deriving V_i by $X \oplus k_i$, U_i calculates t_i by $y_i \oplus t_i \oplus y_i$. Besides, y_A is shared between

U_A and U_i, so if U_A is not impersonated or compromised by an adversary, $h'_i = h_i$ will hold. However, the authentication of U_A in Type II is achieved by public key signature system.

Next, we will discuss the correctness of group authentication by U_A. U_1 calculates C_1 by $C_1 = r^{x_1} = \xi(g^{r_A x_1})$. After U_i $(2 \leq i \leq N)$ receives C_{i-1}, it calculates C_i according to $C_i = C_{i-1} \times r^{x_i} = \xi(g^{r_A \sum_{t=1}^{i} x_t})$. Therefore, $C_N = \xi(g^{r_A \sum_{t=1}^{N} x_t})$ and when U_A receives C_N, it calculates C'_N as $C'_N = \xi((g^{x_1} \times \cdots \times g^{x_N})^{r_A}) = \xi(g^{r_A \sum_{t=1}^{N} x_t})$. We can see that $C_N = C'_N$ holds. Thus, the mutual authentication is correct for protocols of both types.

4.2 Correctness of ECDLP Based Protocols

The authentication of U_A in Type I and Type II are promised by one-way hash function and public key signature system respectively, the same as illustrated in Sect. 4.1, but the authentication of $\mathbb{U}$ relies on C_N. $C_1 = \xi(x_1 G_r) = \xi(r_A x_1 G)$ and $C_i = C_{i-1} + \xi(x_i G_r) = \xi(r_A \sum_{t=1}^{i} x_t G)$ $(2 \leq i \leq N)$. Thus, $C_N = \xi(r_A \sum_{t=1}^{N} x_t G)$. So after U_A receives the message from U_N, it calculates C'_N as $C'_N = \xi(r_A(G_1 + G_2 + \cdots + G_N)) = \xi(r_A \sum_{t=1}^{N} G_t)$. It is straightforward that C_N equals to C'_N.

5 Security Analysis

In this section, we analyze the security requirements of both DLP and ECDLP based protocols. In this paper, we only consider passive adversaries denoted by E, who can only receive messages on the communication channel and then analyzing them acting as a probabilistic polynomial time Turing machine.

5.1 Security Requirements

Both types of our protocols provide mutual and group authentication, protection against passive adversaries and impersonation attacks. They also satisfy implicit key authentication, forward and backward secrecy. We will use the theory of random oracle (RO) [21] and decisional Diffie-Hellman (DDH) [22] assumption to prove these security requirements. First of all, we will introduce the following two assumptions based on which our security proofs are derived.

Assumption 1 ***DLP based DDH Assumption.*** *Suppose G_p is a cyclic group of order p with generator g. $a, b, c \in [1, |G_p|]$ are randomly generated. Given g^a, g^b and g^c, it is supposed that there is no probabilistic polynomial time algorithm to distinguish g^{ab} and g^c.*

Assumption 2 ***ECDLP based DDH Assumption.*** *Suppose E_G is a secure non-singular elliptic curve with G as its base point and n its order, $a, b, c \in [1, n-1]$ are randomly generated. Given aG, bG and cG, it is supposed that there is no probabilistic polynomial time algorithm to distinguish abG and cG.*

Group authentication. It means that U_A can authenticate all members in user group $\mathbb{U}$ at one time. From the correctness analysis in Sect. 4, it is obvious that both the DLP and ECDLP based protocols can provide group authentication.

Mutual authentication. At the end of each protocol, U_A can authenticate each member U_i in $\mathbb{U}$, and U_i can also authenticate U_A. From both Sect. 4 and the message flows in Sect. 3, we can see that the authentication of $\mathbb{U}$ and U_A can be achieved in step (3) and (4) respectively if the protocol is successfully executed.

Theorem 1. *Based on Assumption 1 and 2, both the DLP and ECDLP based protocols are against passive adversaries, which means:*

(1) E cannot derive any information about y_A from h_i;
(2) E cannot derive any information about ξ from C_i.

Proof. In the first step, we will prove that h_i is secure against passive adversaries. Since y_A is protected by the one-away hash function H, based on the theory of RO in [21], the probability that E derives any useful information about y_i from $H(ID_A \oplus ID_i \oplus y_A \oplus t_i)$ is negligible.

In the second step, we prove that E cannot obtain any information about ξ. In DLP and ECDLP based protocols, C_N is computed by $C_i = \xi(g^{r_A \sum_{t=1}^{i} x_t})$ and $C_i = \xi(r_A \sum_{t=1}^{i} x_t G)$, where C_0 is $\xi(g^{r_A})$ and $\xi(r_A G)$ respectively. Obviously, they are DLP and ECDLP based ElGamal encryption. According to the results in [18], ElGamal encryption is as hard as DDH problem. Thus, based on Assumption 1 and 2, C_i is secure against passive adversaries. □

Theorem 2. *Based on Assumption 1 and 2, and the difficulties of DLP and ECDLP, both the DLP and ECDLP based protocols are against impersonation attacks, which means:*

(1) E cannot forge h_i to impersonate U_A;
(2) E cannot forge C_i to impersonate U_i.

Proof. According to (1) of Theorem 1, we know that E cannot derive any information about y_A. And then based on the robustness of one-way hash function [17,23], it is impossible for E to forge h_i without the knowledge of y_A or t_i.

Next, we will demonstrate that it is impossible for E to forge C_i. According to difficulties of DLP and ECDLP, E cannot derive x_i without compromising U_i. Thus, it generates l_i instead and computes $C'_i = \xi(g^{r_A \sum_{i=1}^{i} l_i})/C'_i = \xi(r_A \sum_{i=1}^{i} l_i G)$ (To simplify our presentation, we will use the symbol "/" to represent "or'.'). Since r_A is only known to U_A, E needs to successfully generate $\sum_{i=1}^{i} l_i$ such that $\xi(g^{r_A \sum_{i=1}^{i} l_i})/\xi(r_A \sum_{i=1}^{i} l_i G)$ equals to $\xi(g^{r_A \sum_{i=1}^{i} x_i})/\xi(r_A \sum_{i=1}^{i} x_i G)$. Again, according to the difficulties of DLP and ECDLP, E cannot deduce $\sum_{i=1}^{N} x_i$. However, it can compute the right value of $\xi(g^{r_A \sum_{i=1}^{i} x_i})/\xi(r_A \sum_{i=1}^{i} x_i G)$, which is contradictory to Assumption 1 and 2. As a result, both the DLP and ECDLP based protocols are against impersonate attacks. □

Theorem 3. *Based on the difficulties of DLP and ECDLP, Assumption 1 and 2, and the security of CRT, both the DLP and ECDLP based protocols can provide implicit key authentication, which means:*

(1) Only U_A and U_i can access to the group session key K_G;
(2) Only U_A and U_i can compute the right session key SK_{Ai};
(3) Only U_i and U_j can compute the right session key SK_{ij}, where $1 \leq i, j \leq N$ and $i \neq j$.

Proof. From the formats of V_i and message flows, we know that K_G is protected by CRT. Therefore, E cannot get K_G without knowing k_i.

As for the session key SK_{Ai}, it is computed by parameters g^{m_i}/m_iG and g^{n_i}/n_iG based Diffie-Hellman key exchange system. g^{n_i}/n_iG is transmitted in plaintext, however, E cannot derive n_i by the difficulties of DLP and ECDLP. So the only way for E to obtain SK_{Ai} is to compute it by both key parameters, but this is contradict to Assumption 1 and 2.

The security of session key SK_{ij} is almost the same as SK_{Ai}, with the exception that both g^{n_i}/n_iG and g^{n_j}/n_jG are exposed to the adversaries. For the same reason as mentioned above, E cannot derive session key SK_{ij}.

As a result, both DLP and ECDLP based protocols are proved to provide implicit key authentication. □

Theorem 4. *Based on the difficulties of DLP and ECDLP, Assumption 1 and 2, both the DLP and ECDLP based protocols can provide forward secrecy, which means: the exposure of session key $SK_{Ai,r}$ or $SK_{ij,r}$ in session s_r will not lead to the exposure of session key $SK_{Ai,t}$ or $SK_{ij,t}$ in session s_t, where $1 \leq t < r$.*

Proof. Suppose that all parameters specified to session s_r have been exposed to an adversary E. Here, parameters specified to a session refer to those newly generated in this session, and will be expired when this session finishes, not including those shared in all sessions. Let $g^{m_{i,r}}/m_{i,r}G$ and $g^{n_{i,r}}/n_{i,r}G$ be the key parameters in session s_r, $g^{m_{i,t}}/m_{i,t}G$ and $g^{n_{i,t}}/n_{i,t}G$ be the key parameters in session s_t. Session keys $SK_{Ai,r} = g^{m_{i,r}n_{i,r}}/m_{i,r}n_{i,r}G$ and $SK_{ij,r} = g^{n_{i,r}n_{j,r}}/n_{i,r}n_{j,r}G$ are exposed to E. Since m_i and n_i for each session are randomly generated, and those in a different session cannot be deduced by them. Consequently, even though $SK_{Ai,r}$ and $SK_{ij,r}$ are exposed, $m_{i,t}$, $n_{i,t}$ and $n_{j,t}$ are still unknown to E, and only $g^{m_{i,t}}$, $g^{n_{i,t}}$ and $g^{n_{j,t}}$ are known. Based on the difficulties of DLP and ECDLP, E cannot get $m_{i,t}$, $n_{i,t}$ or $n_{j,t}$ from them. And also by Assumption 1 and 2, we know E cannot compute either $g^{m_{i,t}n_{i,t}}/m_{i,t}n_{i,t}G$ or $g^{n_{i,t}n_{j,t}}/n_{i,t}n_{j,t}G$. As a result,both the DLP and ECDLP based protocols can provide forward secrecy. □

Theorem 5. *Based on the difficulties of DLP and ECDLP, Assumption 1 and 2, both the DLP and ECDLP based protocols can provide backward secrecy, which means: the exposure of session key $SK_{Ai,t}$ or $SK_{ij,t}$ in session s_t will not lead to the exposure of session key $SK_{Ai,r}$ or $SK_{ij,r}$ in session s_r, where $1 \leq t < r$.*

Proof. The proof of Theorem 5 is similar to that of Theorem 4. The only difference is that parameters specified to a session s_t cannot be utilized to deduce those in a later session s_r. □

Except all these security requirements discussed on the above, there is one issue worthy explaining. When U_A and all members in $\mathbb{U}$ calculate MAC, the key they utilize is K_G. Thus there is a possibility that any user who has obtained K_G can tamper another user's message and then calculate the right MAC. However, since the purpose of any user in $\mathbb{U}$ is to get authenticated by U_A, we assume that they will not carry out this kind of inside attacks.

6 Comparisons

Except for authentication and key exchanges, another two purposes we want to achieve in our framework are to save both computation and communication costs. According to Eq. (3), we know that the computation cost mainly depends on operation F and $\circ$, denoted by C_F and $C_\circ$ respectively. There are three possibilities. First, if $C_\circ$ is negligible compared with C_F, the computation cost of our framework is $O(1)$ rather than $O(N)$ in the one-to-one mode with respect to C_F. Second, if $C_\circ$ is almost the same as C_F, the computation cost will be $O(N)$ in both modes. The last possibility is the opposite to the first one, and then our framework becomes more time consuming instead of time saving.

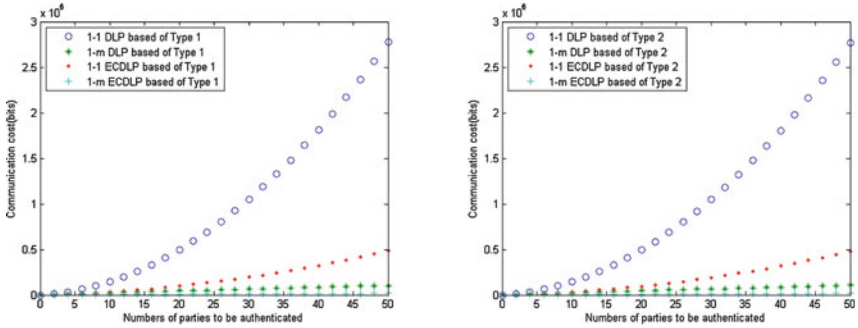
Unlike the complexity of computation cost, our protocol can save some communication cost in general. However, the extent to which it can save depends on many factors, such as which cryptographic primitives are chosen, the length of security parameters and so on. To better compare the communication cost, the general message flows of the traditional one-to-one mode protocol can be simply described as follows.

(1) $U_A \to U_i$: $ID_A, ID_i, y_i \oplus K_G, y_i \oplus t_i, g^{m_i}/m_iG, h_i, C_0, MAC_A$.
  or: $ID_A, SIGN_{SK_A}\{ID_i, y_i \oplus K_G, y_i \oplus t_i, g^{m_i}/m_iG\}, C_0, MAC_A$.
(2) $U_i \to U_A$: $ID_i, ID_A, t'_i, g^{n_i}/n_iG, C_i, MAC_i$.
(3) $U_A \to \mathbb{U}$: ID_A, ID_i, KP_i, MAC'_A.

Here, all the parameters have the same meaning as explained in Table 1. Since the possibility of a bottleneck in communication can mostly happen at U_A, we will only compute the communication cost of U_A. Then in the following,

Table 2. Lengths of parameters (Bits)

	ID	K_G	t_i/t'_i	g^{m_i}/m_iG	h_i	C_N	MAC
DLP based	32	128	128	1,024	160	1,024	160
ECDLP based	32	128	128	160	160	160	160



(a) Communication Cost Comparisons Of Protocols of Type I

(b) Communication Cost Comparisons Of Protocols of Type II

Fig. 3. Communication cost comparisons.

we will show the experiments results about our DLP and ECDLP based protocols in Fig. 3, and the lengths of parameters we use are listed in Table 2. From the figures, we can see that there are big differences between one-to-one (1-1) and one-to-multiple (1-m) modes, and also some differences between DLP and $ECDLP$ based protocols. Suppose the number of user group is N as stated in Table 1, and then the communication cost for protocols in one-to-one mode is $O(N^2)$ but $O(N)$ for one-to-multiple mode protocols. When the number of parties to be authenticated is small, there is not much difference. However, when N increases, the differences will grow fast. Therefore, for systems that have large numbers of users to be authenticated frequently or the computation or communication resources are limited, our framework can gain an obvious advantage.

7 Conclusions

In this paper, we propose a general framework where different cryptographic primitives can be applied to authenticate several users or clients at one time in two scenarios, where authenticators are with or without certificates. In our protocols, mutual authentication can be achieved at the third step. The fourth step for protocols of Type I is optional, since it aims at establishing session keys between different members in group $\mathbb{U}$. By applying our framework, the authenticator can authenticates users or clients in a one-to-multiple mode, which is more effective and thus less time consuming. To demonstrate how our framework works, we give two example, i.e., DLP and ECDLP based protocols. Based on these two examples, we prove that our protocols satisfy certain security requirements, such as against passive and impersonation attacks, and providing implicit key authentication, forward and backward secrecy. In applications of our framework, it is suggested that the certificate-based systems should be the same as the cryptographic primitives to simplify the calculations and also save resources.

References

1. Huiping, J.: Strong password authentication protocols. In: 4th International Conference on Distance Learning and Education (ICDLE), pp. 50–52 (2010)
2. Ghanbarimaman, R., Pour, A.: A new definition of group authentication increasing performance of server calculation. In: International Conference on Information Science and Applications (ICISA), pp. 1–6 (2012)
3. Ren, K., Lou, W., Zhang, Y.: Multi-user broadcast authentication in wireless sensor networks. In: 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON '07), pp. 223–232 (2007)
4. Harn, L.: Group authentication. IEEE Trans. Comput. **62**(9), 1893–1898 (2013)
5. Blake, I., Gao, X., Mullin, R., Vanstone, S., Yaghoobian, T.: The discrete logarithm problem. In: Menezes, A. (ed.) Applications of Finite Fields. The Springer International Series in Engineering and Computer Science, vol. 199, pp. 115–138. Springer, New York (1993)
6. Hankerson, D., Menezes, A.: Elliptic curve discrete logarithm problem. In: Tilborg, H. (ed.) Encyclopedia of Cryptography and Security, pp. 186–189. Springer, New York (2005)
7. Elgamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Inf. Theor **31**(4), 469–472 (1985)
8. Zhao, J., Gu, D.: A security patch for a three-party key exchange protocol. Wuhan Univ. J. Nat. Sci. **15**(3), 242–246 (2010)
9. Zhang, X.L.: Authenticated key exchange protocol in one-round. In: Hua, A., Chang, S.L. (eds.) Algorithms and Architectures for Parallel Processing. LNCS, vol. 5574, pp. 226–233. Springer, Heidelberg (2009)
10. Chaum, D., Van Heyst, E.: Group signatures. In: Proceedings of the 10th Annual International Conference on Theory and Application of Cryptographic Techniques (EUROCRYPT'91), pp. 257–265. Springer, Heidelberg (1991)
11. Rivest, R., Shamir, A., Tauman, Y.: How to leak a secret: theory and applications of ring signatures. In: Goldreich, O., Rosenberg, A., Selman, A. (eds.) Theoretical Computer Science. LNCS, vol. 3895, pp. 164–186. Springer, Heidelberg (2006)
12. Guo, H., Wu, Y., Chen, H., Ma, M.: A batch authentication protocol for v2g communications. In: 4th IFIP International Conference on New Technologies, Mobility and Security (NTMS), pp. 1–5 (2011)
13. Johnson, D., Menezes, A., Vanstone, S.: The elliptic curve digital signature algorithm (ecdsa). Int. J. Inf. Sec. **1**(1), 36–63 (2001)
14. Farah, A., Khali, H.: Joint multiple signature scheme for group-oriented authentication and non-repudiation. In: IEEE GCC Conference, pp. 1–5 (2006)
15. Bellare, M., Neven, G.: Multi-signatures in the plain public-key model and a general forking lemma. In: Proceedings of the 13th ACM conference on Computer and communications security (CCS '06), pp. 390–399. ACM, New York (2006)
16. Yeh, L.Y., Huang, Y.L., Joseph, A., Shieh, S., Tsaur, W.: A batch-authenticated and key agreement framework for p2p-based online social networks. IEEE Trans. Veh. Technol. **61**(4), 1907–1924 (2012)
17. Merkle, Ralph C.: One way hash functions and DES. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 428–446. Springer, Heidelberg (1990)
18. Tsiounis, Y., Yung, M.: On the security of elgamal based encryption. In: Imai, H., Zheng, Y. (eds.) Public Key Cryptography. LNCS, vol. 1431, pp. 117–134. Springer, Heidelberg (1998)

19. Boneh, D.: The decision Diffie-Hellman problem. In: Buhler, J. (ed.) Algorithmic Number Theory. LNCS, vol. 1423, pp. 48–63. Springer, Heidelberg (1998)
20. Chiou, G.H., Chen, W.T.: Secure broadcasting using the secure lock. IEEE Trans. Softw. Eng. **15**(8), 929–934 (1989)
21. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: Proceedings of the 1st ACM conference on Computer and communications security. CCS '93, pp. 62–73. ACM, New York (1993)
22. Boneh, Dan: The decision Diffie-Hellman problem. In: Buhler, Jeremy P. (ed.) ANTS 1998. LNCS, vol. 1423, pp. 48–63. Springer, Heidelberg (1998)
23. Fridrich, J., Goljan, M.: Robust hash functions for digital watermarking. In: Proceedings on the International Conference on Information Technology: Coding and Computing, pp. 178–183 (2000)