Hindawi Publishing Corporation
The Scientific World Journal
Volume 2015, Article ID 275496, 7 pages
http://dx.doi.org/10.1155/2015/275496

Research Article

On Constructing Dynamic and Forward Secure Authenticated Group Key Agreement Scheme from Multikey Encapsulation Mechanism

Iraj Fathirad and John Devlin

Department of Electronic Engineering, La Trobe University, Melbourne, VIC 3086, Australia

Correspondence should be addressed to Iraj Fathirad; i.fathirad@latrobe.edu.au

Received 5 March 2015; Revised 9 August 2015; Accepted 16 August 2015

Academic Editor: Björn Johansson

The approach of instantiating authenticated group key exchange (GAKE) protocol from the multikey encapsulation mechanism (*m*KEM) has an important advantage of achieving classical requirement of GAKE security in one communication round. In spite of the limitations of this approach, for example, lack of forward secrecy, it is very useful in group environments when maximum communication efficiency is desirable. To enrich this *m*KEM-based GAKE construction, we suggest an efficient solution to convert this static GAKE framework into a partially dynamic scheme. Furthermore, to address the associated lack of forward-secrecy, we propose two variants of this generic construction which can also provide a means of forward secrecy at the cost of extra communication round. In addition, concerning associated implementation cost of deploying this generic GAKE construction in elliptic curve cryptosystem, we compare the possible instantiations of this model from existing *m*KEM algorithms in terms of the number of elliptic curve scalar multiplications.

1. Introduction

A reliable and secure shared-key distribution scheme is arguably the most import step toward establishing any cryptographic channel among group of communicating parties. The group key exchange protocol (GKE) allows the members to calculate the shared-key over a public communication medium. An authenticated group key exchange (GAKE) scheme ensures that the resultant shared-key is kept indistinguishable to nonlegitimate peers and provides the participants with resistance against impersonation attack. A recently proposed approach [1] to achieve the classical requirement of authenticated key exchange security [2–4] for group scenario in one communication round is to construct it from the multikey encapsulation mechanism (*m*KEM). We refer to this generic framework by *mk*GAKE model. An *m*KEM [5] is a multipeer cryptographic solution that assumes receivers with long-term certified private/public-key pairs, and enables one entity to generate and efficiently encapsulate the same random session key for multiple receivers. The *mk*GAKE framework is basically constructed by parallel execution of a secure *m*KEM scheme among n parties. While this communicationally efficient GAKE construction provides the participants with basic requirements of key-confidentiality and impersonation resistance, it has two important limitations that can essentially affect the security and functionality of this model. These shortcomings and their undesirable effects are described below:

(1) Lack of forward secrecy (FS): This implies compromising long-term keying materials of peers affects the confidentiality of previously established shared session keys. Since the existing *m*KEM solutions that used as building block of this framework are not FS, the resultant GAKE construction is not FS as well. FS is a desirable feature of a GAKE solution as it ensures the shared-key history remains confidential even after revealing the long-term keying materials of participants.

(2) Inability to provide an efficient solution for dynamic group environments: This implies the participants need to reexecute the GAKE protocol when new members join or existing members leave. It is desirable for a GAKE protocol to provide an efficient mechanism for join/leave operations rather than reexecuting the scheme in dynamic group environment.

Another fundamental factor in analyzing the practicality of this framework is the computational cost of instantiating this generic model. Implementing this framework with the elliptic curve cryptosystem (ECC) can significantly reduce keys/parameters/messages size compared to the non-ECC variants [6]. It is therefore desirable to evaluate the ECC related deployment cost of this generic GAKE construction.

1.1. Our Contribution. In this paper we propose a generic framework to convert the existing static *mk*GAKE model to a partially dynamic scheme which provides a more efficient mechanism in the operation of joining new members to an already established session. In addition, to enrich the existing *mk*GAKE construction, we propose two variants of this model to achieve the important goal of forward secrecy at the cost of an extra communication round. As a further contribution, we evaluate the implementation cost of instantiating this model in ECC from existing *m*KEM schemes.

1.2. Related Works. Most of the group key exchange protocols in literature are based on either DH [11] or Joux [12] algorithms and require multiple rounds to establish the shared-key. Constructing a one-round, yet secure, GAKE is very desirable due to its appealing bandwidth efficiency compared to other multiround solutions. One of the first attempts to formalize one-round GAKE protocols was made in [13] that divides protocols into three different classes:

(1) In the first class, peers are assumed to hold a preshared secret which is impractical and unreasonable in real environments [14].

(2) In the second class one party encrypts its nonce to other participants together with digital signature on its encrypted nonce while other parties send their random values in the clear. An instantiation of such a scheme is given in [15]. It suffers from intensive computational overhead imposed by using digital signature as well as an inherent security concern resulting from unequal distribution of the key exchange responsibilities and giving extreme power to the encryptor party.

(3) In the third class, which equally distributes the power and responsibility between participants, all peers encrypt their nonce to other entities using other participants' certified public-keys. A generic model to efficiently instantiate this class of one-round GAKE scheme from *m*KEM construction is given in [1] (referred to as *mk*GAKE model). So far, this *mk*GAKE model is the only practical one and provably secures one-round implicitly authenticated group key exchange construction in literature to date (while it is theoretically possible to construct a one-round GKE scheme by using multilinear map [16] and then converting it to an implicitly authenticated GAKE scheme in a similar way as MQV [17] or HMQV [18]; but, in spite of some recent improvements in constructing a plausible multilinear map [19, 20], these schemes are still far from being efficiently practical).

1.3. Organization. The next section discusses ECC and *m*KEM schemes and reviews the operation of dynamic GAKE protocols. In this section we also study the existing *m*KEM-based GAKE framework. In Section 3 we present our two variants of *m*KEM-based GAKE model with forward secrecy. In Section 4 we propose a generic framework to convert this existing static GAKE framework to a partially dynamic scheme. Finally, in Section 5 we compare the implementation cost of this framework from the possible ECC translation of existing *m*KEM solutions. Section 6 gives a summary of our work and highlights the important points.

2. Preliminaries

2.1. Elliptic Curve. An elliptic curve E over the prime field of F_q ($characteristic\ (F_q) \neq 2,\ 3$) is defined by a short Weierstrass equation E: $y^2 = x^3 + ax + b$, where the parameters $a, b \in F_q$ are chosen such that $\Delta \neq 4a^3 + 27b^2$ (Δ is the discriminant of the equation). The group of points of E over F_q is denoted by $E(F_q)$ and the order of $E(F_q)$ is indicated by $\#E(F_q)$. An elliptic curve is described by set of parameters (q, a, b, P, p, h), where q specifies the finite field of F_q, a and b are coefficient of E, $P = (x_P, y_P)$ is the generator of a cyclic subgroup of $\langle P\rangle \subset E(F_q)$ of prime order p, and $h = \#E(F_q)/p$ is the cofactor of elliptic curve. Elliptic curve DH assumptions are described as follows:

(i) A Diffie-Hellman (DH) tuple in G is $(P, xP, yP, zP) \in G$ for some $x, y, z \in Z_q^*$ satisfying $z = xy \bmod q$.

(ii) *Computational Diffie-Hellman (CDH) problem*: given any three elements from the four elements in a DH tuple compute the remaining element.

(iii) *Decision Diffie-Hellman (DDH) problem*: given $P, xP, yP, zP \in G$, decide if it is a valid DH tuple or not (if $zP = xyP$).

(iv) *Hashed Decision Diffie-Hellman (HDDH) problem*: given $P, xP, yP \in G$ and h, decide if $h = H(xyP)$, where H is a target collision resistant Hash function.

2.2. mKEM. An *m*KEM scheme allows a peer to efficiently encapsulate a single session key to n parties. A typical *m*KEM scheme is presented by $(\mathcal{G}_m, \mathcal{E}_m, \mathcal{D}_m)$ tuples and consists of three core algorithms: private/public-key generation ($\mathcal{G}_m$), key encapsulation ($\mathcal{E}_m$), and key decapsulation ($\mathcal{D}_m$). The probabilistic algorithm of $\mathcal{G}_m$ takes domain parameters and generates public/private-key pairs (pk_i, sk_i). The probabilistic algorithm of $\mathcal{E}_m$ takes set of public-keys $pk = \{pk_1, \ldots, pk_n\}$

of receivers and returns the encapsulation pair (K, C), where $C = \{c_1, \ldots, c_n\}$ and c_i is encapsulation of K with pk_i. The deterministic algorithm of $\mathscr{D}_m$ takes private-key sk_i and the encapsulation C and outputs K. For a mKEM scheme to be secure it is required to be sound, which means, for all key pairs (pk_i, sk_i) generated by $\mathscr{G}_m(\gamma)$ and all encapsulation pair (K, C) generated by $\mathscr{E}_m(\gamma, pk)$, we assume all possible range of K is generated by $\mathscr{D}_m(\gamma, C, sk_i)$.

2.3. Dynamic Group Key Exchange. The GKE algorithms are divided into two groups of *static* and *dynamic* in terms of their capability to reform the session key with updated group membership. In static GKE the number of peers remains constant during the session whereas in dynamic GKE participants are allowed to join or leave the session at any time during the active session. A typical dynamic GKE consists of three algorithms, namely, shared-key establishment scheme, *join* operation, and *leave* operation. The shared-key establishment scheme operates the same as typical static GKE scheme and allows n parties to securely calculate confidential shared-key. The *join* operation allows new member to jointly establish new key with existing members in the way that the new member should not be able to extract the previously established session keys between those peers. The *leave* operation removes one of the members from the existing session and allows the remaining members to calculate a fresh key for the session. The leaving group members should be capable of calculating or distinguishing updated session key. Whilst it is possible to convert any static GKE to a dynamic GKE by reexecuting the static GKE with updated members, it is desirable for a GKE protocol to provide more efficient solution for join/leave operations rather than trivial approach of reexecuting the GKE scheme in dynamic environment.

2.4. mKEM-Based One-Round GAKE Construction (mkGAKE Model). The generic model proposed by Gorantla et al. in [1] provides a framework to construct a one-round implicitly authenticated group key exchange (GAKE) from an mKEM scheme. Consider set of parties $\mathscr{U} = \{u_1, \ldots, u_n\}$ as participants in the GAKE scheme, where $u_{i\in[1,n]}$ is the identity of a participant and $\mathscr{U}$ is set of identities of all parties. This generic model assumes an IND-CCA secure $m\text{KEM}(\mathscr{G}_m, \mathscr{E}_m, \mathscr{D}_m)$ as the core algorithm and is designed to let the members of $\mathscr{U}$ establish a shared session key through parallel execution $m\text{KEM}(\mathscr{G}_m, \mathscr{E}_m, \mathscr{D}_m)$. Since mKEM guarantees to the sender that only the legitimate receiver can decapsulate the session key, this generic model constructed from parallel execution of mKEM among multiple participants can provide all parties with implicit authentication on computed symmetric-key. This model consists of four phases as shown below:

(i) Initiation:

$$\begin{aligned} &(u_i \in \mathscr{U}) : (sk_i, pk_i) \longleftarrow \mathscr{G}_{m\text{KEM}}(\mathbb{D}), \\ &pk = \{pk_1, \ldots, pk_n\} \text{ is known to all peers.} \end{aligned} \tag{1}$$

(ii) Computation:

$$\begin{aligned} &(u_i \in \mathscr{U}) : (K_i, C_i) \\ &\quad \longleftarrow \mathscr{E}_{m\text{KEM}}\left(\left\{pk_j \mid pk_j \in pk;\ 1 \le j \ne i \le n\right\}\right). \end{aligned} \tag{2}$$

(iii) Communication:

$$(u_i \in \mathscr{U}) \text{ broadcasts } (u_i, c_i) \text{ to } \mathscr{U} \setminus \{u_i\}. \tag{3}$$

(iv) Key calculation:

$$\begin{aligned} &(u_i \in \mathscr{U}) \text{ calculate the shared-key as follows} \\ &\quad K_j \\ &1 \le j \ne i \le n \\ &\quad \longleftarrow \mathscr{D}_{m\text{KEM}}\left(sk_i, \left\{C_j \mid 1 \le j \ne i \le n\right\}\right), \\ &\text{sid} \longleftarrow (C_1 \parallel \cdots \parallel C_n \parallel \mathscr{U}), \\ &K \longleftarrow f_{K_1}(\text{sid}) \oplus \cdots \oplus f_{K_n}(\text{sid}). \end{aligned} \tag{4}$$

In the *Initiation* phase, each GAKE participant $u_i \in \mathscr{U}$ executes $\mathscr{G}_{m\text{KEM}}(\mathbb{D})$ to obtain private/public-key pairs of (sk_i, pk_i), and authentic set of public-keys $pk = \{pk_1, \ldots, pk_n\}$ is known to all peers. In the *Computation* phase, each $u_i \in \mathscr{U}$ executes mKEM encapsulation algorithm with other participants' public-key to obtain the symmetric-key and encapsulation pair. In the *Communication* phase each $u_i \in \mathscr{U}$ broadcasts its computed encapsulation together with its id to all other peers. Finally, in the *Key calculation* phase each $u_i \in \mathscr{U}$ executes the mKEM decapsulation algorithm on each of the incoming encapsulations using its private-key to obtain $(n-1)$ number of the symmetric-keys. Then, set sid to be the concatenation of all the incoming and outgoing exchanged messages $\text{sid} \leftarrow (C_1 \parallel \cdots \parallel C_n \parallel \mathscr{U})$, where $\mathscr{U}$ is the set of identities of all the users. Finally, sid and decapsulated keys are fed to a pseudorandom function to calculate the session key.

3. Two-Round GAKE with Forward Secrecy

The generic one-round mKEM-based framework cannot provide forward secrecy, but it can be extended to a two-round unauthenticated scheme to achieve this additional goal. In this approach, the authenticated and certified long-term private/public-key pairs are replaced with ephemeral key pairs. In the two-round variance, the participants execute the mKEM in parallel with on-demand generated ephemeral keys. Using ephemeral and uncertified asymmetric-keys will result in a GKE protocol without an implicit authentication property. In this case an adversary can impersonate any honest participant to other peers by replacing the ephemeral private/public values and resultant protocol is only secure in the presence of a passive adversary. To provide the GKE protocol with authentication, one of the two following approaches may be adopted.

3.1. Using Digital Signature in the First Round. In this approach the peer $u_i \in \mathcal{U}$ is assumed to hold a certified long-term private/public-key pair of $(\mathcal{SK}_i, \mathcal{PK}_i)$ corresponding to the employed digital signature scheme. The signing/verification algorithms of the corresponding digital signature scheme are denoted by (Sign, Verify). The key exchange procedure is carried out in two rounds as shown below.

Round 1. Peer u_i runs the $\mathcal{G}_{m\text{KEM}}$ function to obtain ephemeral private/public-key pair of (sk_i, pk_i) and then use $\mathcal{SK}_i$ to compute the digital signature on pk_i. Then, u_i broadcasts the signature together with its id and ephemeral public-key of pk_i to the other users. The generic framework for the first round interaction is shown below:

(i) Setup:

$$\left(u_i \in \mathcal{U}\right): \text{obtains } \frac{\text{long-term signing}}{\text{verification keys of } \left(\mathcal{SK}_i, \mathcal{PK}_i\right)}. \quad (5)$$

The verification public-keys $\mathcal{PK} = \{\mathcal{PK}_1, \ldots, \mathcal{PK}_n\}$ are known to all peers.

(ii) Initiation:

$$\left(u_i \in \mathcal{U}\right) : \left(sk_i, pk_i\right) \longleftarrow \mathcal{G}_{m\text{KEM}}\left(\mathbb{D}\right). \quad (6)$$

(iii) Signature:

$$\left(u_i \in \mathcal{U}\right) : \sigma_i \longleftarrow \text{Sign}_{\mathcal{SK}_i}\left(u_i, pk_i\right). \quad (7)$$

(iv) Communication 1:

$$\left(u_i \in \mathcal{U}\right) \text{ broadcasts } \left(u_i, \sigma_i, pk_i\right) \text{ to } \mathcal{U} \setminus \{u_i\}. \quad (8)$$

Round 2. In the second round, other peers verify the authenticity of pk_i by validating the received signature using the publicly available certified verification key of $\mathcal{PK}_i$ and then run the one-round protocol with authentic ephemeral public-keys. The generic framework for the second-round interaction of this approach is shown below:

(i) Verification:

$$\text{For } 1 \le j \ne i \le n \text{ peer } \left(u_i \in \mathcal{U}\right) \text{ Check } \text{Verify}_{\mathcal{PK}_j}\left(\left(u_j, pk_j\right), \sigma_j\right) = 1. \quad (9)$$

(ii) Computation:

$$\left(u_i \in \mathcal{U}\right) : \left(K_i, C_i\right) \longleftarrow \mathcal{E}_{m\text{KEM}}\left(\left\{pk_j \mid pk_j \in pk;\ 1 \le j \ne i \le n\right\}\right). \quad (10)$$

(iii) Communication 2:

$$\left(u_i \in \mathcal{U}\right) \text{ broadcasts } \left(u_i, c_i\right) \text{ to } \mathcal{U} \setminus \{u_i\}. \quad (11)$$

(iv) Key calculation:

$$\left(u_i \in \mathcal{U}\right) \text{ calculate the shared-key as follows}$$

$$K_j \quad 1 \le j \ne i \le n \longleftarrow \mathcal{D}_{m\text{KEM}}\left(sk_i, \left\{C_j \mid 1 \le j \ne i \le n\right\}\right),$$

$$\text{sid} \longleftarrow \left(C_1 \parallel \cdots \parallel C_n \parallel \mathcal{U}\right),$$

$$K \longleftarrow f_{K_1}(\text{sid}) \oplus \cdots \oplus f_{K_n}(\text{sid}). \quad (12)$$

3.2. Using Digital Signature in the Second Round. A variant of this approach is (also) suggested in [21] and the core idea is originally borrowed from [22]. In this framework, peer $u_i \in \mathcal{U}$ is assumed to have a pair of certified long-term signing/verification key pair of $(\mathcal{SK}_i, \mathcal{PK}_i)$ corresponding to the employed digital signature scheme. The generic framework for the first-round interaction is shown below.

Round 1. In the first round, peer u_i runs the $g_{m\text{KEM}}$ function to obtain ephemeral private/public-key pair of (sk_i, pk_i) and broadcast it to other peers:

(i) Setup:

$$\left(u_i \in \mathcal{U}\right): \text{obtains } \frac{\text{long-term signing}}{\text{verification keys of } \left(\mathcal{SK}_i, \mathcal{PK}_i\right)}. \quad (13)$$

The verification public-keys $\mathcal{PK} = \{\mathcal{PK}_1, \ldots, \mathcal{PK}_n\}$ are known to all peers.

(ii) Initiation:

$$\left(u_i \in \mathcal{U}\right) : \left(sk_i, pk_i\right) \longleftarrow \mathcal{G}_{m\text{KEM}}\left(\mathbb{D}\right). \quad (14)$$

(iii) Communication:

$$\left(u_i \in \mathcal{U}\right) \text{ broadcasts } \left(pk_i\right) \text{ to } \mathcal{U} \setminus \{u_i\}. \quad (15)$$

Round 2. In the second round, each $u_i \in \mathcal{U}$ executes mKEM encapsulation algorithm with other participants' public-keys to obtain the symmetric-key and encapsulation pair. To provide the authentication property, u_i uses $\mathcal{SK}_i$ to compute digital signature on session key encapsulation C_i concatenated with ephemeral public-keys in the system and broadcast signature and C_i to other participants. Other peers verify the authenticity of received encapsulation of C_i (and corresponding embedded key) by validating the received signature using the publicly available certified verification key of $\mathcal{PK}_i$. After validating the authenticity of all received

encapsulations, each peer extracts the embedded-keys by using mKEM decapsulation algorithm and finally computes the shared session key. The generic framework for the second round interaction is shown below:

(i) Computation:

$$\begin{aligned}&(u_i \in \mathcal{U}) : (K_i, C_i)\\ &\quad \longleftarrow \mathscr{E}_{m\mathrm{KEM}}\left(\left\{pk_j \mid pk_j \in pk;\ 1 \le j \ne i \le n\right\}\right).\end{aligned} \tag{16}$$

(ii) Signature:

$$(u_i \in \mathcal{U}) : \sigma_i \longleftarrow \mathrm{Sign}_{\mathcal{SK}_i}(C_i \mid pk_1 \mid \cdots \mid pk_n). \tag{17}$$

(iii) Communication:

$$(u_i \in \mathcal{U}) \text{ broadcasts } (u_i, \sigma_i, C_i) \text{ to } \mathcal{U} \setminus \{u_i\}. \tag{18}$$

(iv) Verification:

$$\begin{aligned}&\text{For } 1 \le j \ne i \le n \text{ peer } (u_i \in \mathcal{U}) \text{ Check}\\ &\mathrm{Verify}_{\mathcal{PK}_j}\left(\left(u_j, C_j \mid pk_1 \mid \cdots \mid pk_n\right), \sigma_j\right) = 1.\end{aligned} \tag{19}$$

(v) Key calculation:

$$\begin{aligned}&(u_i \in \mathcal{U}) \text{ calculate the shared-key as follows}\\ &\quad K_j\\ &1 \le j \ne i \le n\\ &\quad \longleftarrow \mathscr{D}_{m\mathrm{KEM}}\left(sk_i, \left\{C_j \mid 1 \le j \ne i \le n\right\}\right),\\ &\mathrm{sid} \longleftarrow (C_1 \parallel \cdots \parallel C_n \parallel \mathcal{U}),\\ &K \longleftarrow f_{K_1}(\mathrm{sid}) \oplus \cdots \oplus f_{K_n}(\mathrm{sid}).\end{aligned} \tag{20}$$

3.3. Security Analysis. The provided security of both approaches relies on the security of the underlying digital signature scheme. Both approaches assume each peer $u_i \in \mathcal{U}$ possesses a certified long-term private/public-key pair of $(\mathcal{SK}_i, \mathcal{PK}_i)$ corresponding to the employed digital signature scheme. The relevant private-key, which is tasked to sign either ephemeral public-key (first approach) or encapsulation of the session key (second approach), should be kept secret as revealing this key to a potential adversary would result in revealing the session key. If we assume the employed signature scheme is secure against existential forgery under an adaptive chosen message attack, and the corresponding signing private-keys are kept secret from potential adversaries, then we can conclude both approaches are forward-secure authenticated group key exchange schemes. In fact, both approaches use a hierarchy of signatures where the first authenticated exchange authenticates the next exchange.

It should be noted that the ephemeral session keys are independent of the long-term keying materials, and the long-term keys are only tasked to authenticate the session key and not to take a role in the calculation of these keys. Thus, if an adversary manages to compromise a long-term keying material of any participating peers in a random session, he/she cannot reveal any information about the ephemeral keys of previous sessions in which the corrupted party has been participating in the past. However, the future session keys will not be secure against this adversary as he/she can fake the corrupted party and fool any other party(ies) to enter session key calculation phase with her/him. It should be noted that a forward-secure group key exchange is expected to keep the previous session keys unaccessible, not the future keys. Based on this notion, both of the described approaches are forward-secure key exchange schemes. Note that while the first approach (Section 3.1) is basically simpler and more convenient, the second method (Section 3.2) is stronger as it provides mean of mutual authentication on all the session-related ephemeral values.

4. Achieving Dynamic Group Operation

While it is possible to construct a dynamic protocol from this GAKE model by reexecuting (from scratch) the scheme in *join* or *leave* procedure, we propose a solution to perform the *join* operation more efficiently. The *leave* operation still requires the member to reexecute the protocol.

Consider a scenario where a new member u_z, with knowledge of domain parameters $\mathbb{D}$, KDF, and f, decides to join an ongoing session between $\mathcal{U}$ parties with the shared session key of K_S. Peer u_z is required to run the $\mathscr{G}_{m\mathrm{KEM}}(\mathbb{D})$ function to obtain the private/public-key pair (sk_z, pk_z). The members of $\mathcal{U}$ are required to have access to the certified public-key of u_z. The members of $\mathcal{U}$ also should have access to a secure symmetric encryption algorithm denoted by (E, D). To join the ongoing session, each of $\mathcal{U}$ members together with u_z should follow the corresponding procedure as described below.

Peer u_z. This new member executes the one-round protocol with respect to $\mathcal{U}$ public-keys of $pk = \{pk_1, \ldots, pk_n\}$:

(i) Initiation:

$$\begin{aligned}&(sk_z, pk_z) \longleftarrow \mathscr{G}_{m\mathrm{KEM}}(\mathbb{D}),\\ &pk_z \text{ is known to all peers.}\end{aligned} \tag{21}$$

(ii) Computation:

$$(u_z) : (K_z, C_z) \longleftarrow \mathscr{E}_{m\mathrm{KEM}}(\{pk_1, \ldots, pk_n\}). \tag{22}$$

(iii) Communication:

$$(u_z) \text{ broadcasts } (u_z, C_z) \text{ to } \mathcal{U}. \tag{23}$$

(iv) Key calculation:

$$\begin{aligned}&(u_z) \text{ calculate the shared-key as follows}\\ &\quad K_i\\ &1 \le i \le n \longleftarrow \mathscr{D}_{m\mathrm{KEM}}\left(sk_z, \left\{C_i \mid 1 \le i \le n\right\}\right),\\ &\mathrm{sid} \longleftarrow (\mathscr{C}_1 \parallel \cdots \parallel \mathscr{C}_n \parallel C_1 \parallel \cdots \parallel C_n \parallel C_z \parallel \mathcal{U} \parallel u_z),\\ &K \longleftarrow f_{K_1}(\mathrm{sid}) \oplus \cdots \oplus f_{K_n}(\mathrm{sid}) \oplus f_{K_z}(\mathrm{sid}).\end{aligned} \tag{24}$$

Table 1: Efficiency comparison of GAKE construction from different mKEMs.

Scheme	Assumption	Initiation			Key exchange performance for n participants							
					Computation			Key calculation			Total ≈SM	Total KDF/Hash
		SM	MSM	Key size pk/sk	SM	MSM	KDF/Hash	SM	MSM	KDF/Hash		
PSEC [7]	CDH	1	—	2/1	n	—	$n+1/-$	$2n-2$	—	$2n-2/-$	$3n-2$	$3n-1/-$
ELGamal [5]	CDH	1	—	2/1	n	—	1/1	$2n-2$	—	$n-1/n-1$	$3n-2$	n/n
CS98 [8]	DDH	1	2	5/5	$n+1$	$n-1$	$1/n-1$	$n-1$	$n-1$	$n-1/n-1$	$4.78n-2.78$	$n/2n-2$
HK07 [9]	HDDH	3	—	4/3	2	$n-1$	$-/1$	—	$n-1$	$-/n-1$	$2.78n-0.78$	$-/n$
HTAS-CKS [10]	HDDH	—	3	5/5	2	$2n-2$	$-/2$	$2n-2$	$2n-2$	$-/2n-2$	$7.56n-5.56$	$-/2n$
HTAS-HK [10]	HDDH	2	—	4/4	2	$n-1$	$-/2$	—	$n-1$	$-/2n-2$	$2.78n-0.78$	$-/2n$

Peer $u_i \in \mathcal{U}$. Each member of $\mathcal{U}$ executes the one-round protocol with respect to u_z public-key of pk_z and use a symmetric encryption scheme with already established session key to distribute the new keys among themselves:

(i) Computation:

$$\begin{aligned}(u_i \in \mathcal{U}) : (K_i, C_i) &\longleftarrow \mathscr{E}_{m\mathrm{KEM}}(pk_z),\\ \mathscr{C}_i &\longleftarrow E_{K_s}(K_i).\end{aligned} \tag{25}$$

(ii) Communication:

$$(u_i \in \mathcal{U}) \text{ broadcasts } (u_i, \mathscr{C}_i, C_i) \text{ to } \{\{\mathcal{U} \setminus u_i\} \cup u_z\}. \tag{26}$$

(iii) Key calculation:

$(u_i \in \mathcal{U})$ calculate the shared-key as follows

$$\begin{aligned}&\overset{K_j}{1 \le j \ne i \le n} \longleftarrow D_{K_s}(\{\mathscr{C}_i \mid 1 \le j \ne i \le n\}),\\ &K_z \longleftarrow \mathscr{D}_{m\mathrm{KEM}}(sk_i, C_z),\\ &\mathrm{sid} \longleftarrow (\mathscr{C}_1 \parallel \cdots \parallel \mathscr{C}_n \parallel C_1 \parallel \cdots \parallel C_n \parallel C_z \parallel \mathcal{U} \parallel u_z),\\ &K \longleftarrow f_{K_1}(\mathrm{sid}) \oplus \cdots \oplus f_{K_n}(\mathrm{sid}) \oplus f_{K_z}(\mathrm{sid}).\end{aligned} \tag{27}$$

It should be noted that in reexecuting the GAKE protocol from scratch each member needs to execute the associated mKEM protocol with public-keys of all existing and new members. However, with the help of the proposed framework, each of existing members of $\mathcal{U}$ executes the associated mKEM protocol, in the joining of new member(s), only with public-key(s) of that (those) member(s) and does not include the public-keys of other existing members $\{\mathcal{U} \setminus u_i\}$. Considering the expensive computational cost of mKEM schemes which is dependent on the number of inputted public-keys, this framework can significantly reduce the associated computational overhead in dynamic group environments. While this construction results in better efficiency compared to rerunning the algorithm in joining a new member, the security of this scheme solely depends on the security of the employed symmetric encryption scheme and security of the generic mKEM-based GAKE model (Section 2.4). The joining new peer of u_z executes the mKEM-based GAKE model with the existing members; thus, the security of the calculated key with this node is the same as the generic framework. Furthermore, other nodes of $u_i \in \mathcal{U}$ execute the mKEM-based GAKE model with the new peer of u_z and, in the meantime, distribute their ephemeral values among themselves through a CCA2 symmetric encryption scheme; thus, the security of calculated session key with these peers relies on the security of the mKEM-based GAKE framework and employed symmetric encryption scheme, combined.

5. Efficiency Comparison of Instantiating GAKE Model from Different KEMs

In Table 1 we compare the efficiency of instantiating the mkGAKE model from existing provably secure mKEM schemes. The table compares the computation cost of such constructions in terms of number of associated EC point scalar multiplications which is denoted by SM. The SM calculation refers to computing kP where k is an integer and P is an EC point. The multiscalar multiplication denoted by MSM refers to computing $\sum k_i P_i$. We found it easier and more consistent to represent the computational efficiency of different schemes by a single element of SM. However, since many factors contribute to computation of various MSM cases, then it is very difficult to precisely describe the computation cost of MSM in terms of calculation cost of SM. One approach to roughly estimate this relation, as described in [23], is by considering the unsigned binary representation of scalars and calculate MSM with a sliding window technique. An estimation from this approach is described in [24] and in windows size of 2 and bit-length of 256 it is assumed that one MSM calculation is roughly equal to 1.39 SM calculation. It should be noted that this optimistic estimation enables us to conveniently compare the computation efficiency of different EC-based schemes in a unified system.

6. Conclusion

Through this contribution, we propose an efficient and practical generic framework to convert static mKEM-based

GAKE construction into a partially dynamic scheme. Our framework provides a more efficient solution for the join operation rather than the naive approach of reexecuting the original GAKE model with updated memberships. Furthermore, in order to enrich existing *m*KEM-based GAKE framework, we propose two variants of this generic model which can also provide a means of forward secrecy at the cost of an extra communication round. Finally, to evaluate the computational cost of deploying this generic model in elliptic curve cryptosystem, we compared the associated EC-related calculation cost of possible instantiations of this model from existing *m*KEM algorithms.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] M. C. Gorantla, C. Boyd, J. M. G. Nieto, and M. Manulis, "Generic one round group key exchange in the standard model," in *Information, Security and Cryptology—ICISC 2009*, pp. 1–15, Springer, Berlin, Germany, 2010.

[2] E. Bresson, O. Chevassut, D. Pointcheval, and J.-J. Quisquater, "Provably authenticated group Diffie-Hellman key exchange," in *Proceedings of the 8th ACM Conference on Computer and Communications Security*, pp. 255–264, 2001.

[3] E. Bresson, O. Chevassut, and D. Pointcheval, "Dynamic group Diffie-Hellman key exchange under standard assumptions," in *Advances in Cryptology—EUROCRYPT 2002*, vol. 2332 of *Lecture Notes in Computer Science*, pp. 321–336, Springer, Berlin, Germany, 2002.

[4] E. Bresson, O. Chevassut, and D. Pointcheval, "Provably authenticated group Diffie-Hellman key exchange—the dynamic case," in *Advances in Cryptology—ASIACRYPT 2001*, pp. 290–309, Springer, 2001.

[5] N. P. Smart, "Efficient key encapsulation to multiple parties," in *Security in Communication Networks*, pp. 208–219, Springer, 2005.

[6] V. Gupta, S. Gupta, S. Chang, and D. Stebila, "Performance analysis of elliptic curve cryptography for SSL," in *Proceedings of th ACM Workshop on Wireless Security*, pp. 87–94, September 2002.

[7] V. Shoup, *A Proposal for an ISO Standard for Public Key Encryption (Version 2.1)*, vol. 112, IACR E-Print Archive, 2001.

[8] R. Cramer and V. Shoup, "Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack," *SIAM Journal on Computing*, vol. 33, no. 1, pp. 167–226, 2003.

[9] D. Hofheinz and E. Kiltz, "Secure hybrid encryption from weakened key encapsulation," in *Advances in Cryptology-CRYPTO 2007*, pp. 553–571, Springer, 2007.

[10] H. Hiwatari, K. Tanaka, T. Asano, and K. Sakumoto, "Multi-recipient public-key encryption from simulators in securitu proofs," in *Information Security and Privacy*, pp. 293–308, Springer, 2009.

[11] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[12] A. Joux, "A one round protocol for tripartite Diffie-Hellman," in *Algorithmic Number Theory*, pp. 385–393, Springer, 2000.

[13] C. Boyd, "Towards a classification of key agreement protocols," in *Proceedings of the 8th IEEE Computer Security Foundations Workshop*, pp. 38–43, IEEE, June 1995.

[14] C. Boyd, "On key agreement and conference key agreement," in *Information Security and Privacy*, vol. 1270 of *Lecture Notes in Computer Science*, pp. 294–302, Springer, Berlin, Germany, 1997.

[15] C. Boyd and J. M. G. Nieto, "Round-optimal contributory conference key agreement," in *Public Key Cryptography—PKC 2003*, vol. 2567 of *Lecture Notes in Computer Science*, pp. 161–174, Springer, Berlin, Germany, 2002.

[16] D. Boneh and A. Silverberg, "Applications of multilinear forms to cryptography," *Contemporary Mathematics*, vol. 324, pp. 71–90, 2003.

[17] L. Law, A. Menezes, M. Qu, J. Solinas, and S. Vanstone, "An efficient protocol for authenticated key agreement," *Designs, Codes and Cryptography*, vol. 28, no. 2, pp. 119–134, 2003.

[18] H. Krawczyk, "HMQV: a high-performance secure Diffie-Hellman protocol," in *Advances in Cryptology—CRYPTO 2005*, vol. 3621 of *Lecture Notes in Computer Science*, pp. 546–566, Springer, Berlin, Germany, 2005.

[19] J.-S. Coron, T. Lepoint, and M. Tibouchi, "Practical multilinear maps over the integers," in *Advances in Cryptology—CRYPTO 2013*, vol. 8042 of *Lecture Notes in Computer Science*, pp. 476–493, Springer, Berlin, Germany, 2013.

[20] C. Gentry, S. Gorbunov, and S. Halevi, "Graded multilinear maps from lattices," Tech. Rep. 2014/645, Cryptology ePrint Archive, 2014, http://eprint.iacr.org/.

[21] M. C. Gorantla, *Design and analysis of group key exchange protocols [Ph.D. thesis]*, Queensland University of Technology, Brisbane, Australia, 2010.

[22] J. Katz and M. Yung, "Scalable protocols for authenticated group key exchange," in *Advances in Cryptology—CRYPTO 2003*, pp. 110–125, Springer, 2003.

[23] R. M. Avanzi, "The complexity of certain multi-exponentiation techniques in cryptography," *Journal of Cryptology*, vol. 18, no. 4, pp. 357–373, 2005.

[24] J. Baek, W. Susilo, J. K. Liu, and J. Zhou, "A new variant of the Cramer-Shoup KEM secure against chosen ciphertext attack," in *Applied Cryptography and Network Security*, vol. 5536 of *Lecture Notes in Computer Science*, pp. 143–155, Springer, Berlin, Germany, 2009.