Asymmetric Group Key Agreement

Qianhong Wu[1,2], Yi Mu[3], Willy Susilo[3], Bo Qin[1,4] and Josep Domingo-Ferrer[1]

[1] Universitat Rovira i Virgili, Dept. of Comp. Eng. and Maths
UNESCO Chair in Data Privacy, Tarragona, Catalonia
{qianhong.wu,bo.qin,josep.domingo}@urv.cat
[2] Key Lab. of Aerospace Information Security and Trusted Computing
Ministry of Education, School of Computer, Wuhan University, China
[3] Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Australia
{ymu,wsusilo}@uow.edu.au
[4] Dept. of Maths, School of Science, Xi'an University of Technology, China

Abstract. A group key agreement (GKA) protocol allows a set of users to establish a common secret via open networks. Observing that a major goal of GKAs for most applications is to establish a confidential channel among group members, we revisit the group key agreement definition and distinguish the conventional (*symmetric*) group key agreement from *asymmetric group key agreement* (ASGKA) protocols. Instead of a common secret key, only a shared encryption key is negotiated in an ASGKA protocol. This encryption key is accessible to attackers and corresponds to different decryption keys, each of which is only computable by one group member. We propose a generic construction of *one-round* ASGKAs based on a new primitive referred to as aggregatable signature-based broadcast (ASBB), in which the public key can be simultaneously used to verify signatures and encrypt messages while any signature can be used to decrypt ciphertexts under this public key. Using bilinear pairings, we realize an efficient ASBB scheme equipped with useful properties. Following the generic construction, we instantiate a one-round ASGKA protocol *tightly* reduced to the decision Bilinear Diffie-Hellman Exponentiation (BDHE) assumption in the standard model.

1 Introduction

Many complex cryptosystems rely on the existence of a confidential channel among the users. A major goal of key agreement protocols is to establish such a channel for two or more users. Since the inception of the Diffie-Hellman protocol [12] in 1976, it has been an elusive open problem to construct a *one-round* group key agreement protocol *from scratch*. A *round* means that each party sends one message and can broadcast simultaneously. A key agreement protocol is said to be *from scratch* if each participant does not hold any secret values prior to the execution of the protocol. Each type of long-term-key free protocols can only provide security against passive attackers, but they are the basis to build

advanced protocols against more powerful attackers. We concentrate on new key agreement protocols from scratch in this report.

1.1 Our Contribution

This paper investigates a close variation of the above mentioned problem of one-round group key agreement protocols and focuses on *"how to establish a confidential channel from scratch for multiple parties in one round"*. We provide a short overview of some new ideas to solve this variation.

Asymmetric GKA. Observe that a major goal of GKAs for most applications is to establish a confidential broadcast channel among the group. We investigate the potentiality to establish this channel in an asymmetric manner in the sense that the group members merely negotiate a common encryption key (accessible to attackers) but hold respective secret decryption keys. We introduce a new class of GKA protocols which we name *asymmetric group key agreements* (ASGKAs), in contrast to the conventional GKAs. A trivial solution is for each member to publish a public key and withhold the respective secret key, so that the final ciphertext is built as a concatenation of the underlying individual ones. However, this trivial solution is highly inefficient: the ciphertext increases linearly with the group size; furthermore, the sender has to keep all the public keys of the group members and separately encrypt for each member. We are interested in nontrivial solutions that do not suffer from these limitations.

Aggregatable signature-based broadcast (ASBB). Our proposals rely on a new notion named *aggregatable signature-based broadcast*. In an ASBB scheme, the public key can be simultaneously used to verify signatures and encrypt messages, and any valid signature can be used to decrypt ciphertexts under this public key; furthermore, an ASBB scheme satisfies the *key-homomorphic property* and the *aggregatability* property. The key-homomorphic property means that the combination of signatures on the *same message* produces a valid signature of this message under the combination of the corresponding public keys. As a consequence, the combined signature can be used as a decryption key of the new ASBB instance. Aggregatability states that the combination of secure ASBB instances produces a new secure ASBB instance.

Non-trivial one-round ASGKA. We propose a non-trivial one-round ASGKA scheme. Our idea is to generate the public key of an ASBB scheme in a distributed manner, such that only each member can obtain a signature under this public key. These signatures can be used as their respective decryption keys and a confidential channel among the group is established. We build an efficient ASBB scheme from bilinear pairings and prove its security under the decision n-Bilinear Diffie-Hellman exponentiation (n-BDHE) assumption with the help of a random oracle. By following the generic construction and exploiting the randomness in the setup stage, we instantiate a one-round ASGKA protocol and tightly reduce its security to the decision n-BDHE assumption in the standard model (without using random oracles). The proposed one-round ASGKA protocol achieves the confidential channel of a one-round conventional GKA protocol.

Also, our ASGKA proposal has additional advantages, *e.g.*, serving as a public key based broadcast scheme without requiring a dealer.

1.2 One-round Group Key Agreement

A key agreement protocol enables two or more users within an open network to create a common secret key. In what follows we review round-efficient key agreement protocols, especially, one-round protocols. These protocols are unauthenticated and only secure against passive attacks but they are the basis on which protocols with active security can be built. To date, only very few one-round key agreement protocols (excluding their variations) have been found, *i.e.*, the two-party Diffie-Hellman [12] and the tripartite Joux [19] protocols.

The basic Diffie-Hellman protocol [12] is a one-round two-party protocol. After each party publishes one element in a finite cyclic group, two parties can establish a common secret key. If one party fixes its published element as its public key and the other party generates its element dynamically for each session, the Diffie-Hellman protocol implies a public key cryptosystem, *i.e.*, the well-known ElGamal cryptosystem [13].

The Joux protocol [19] is a one-round tripartite protocol. Similarly to the Diffie-Hellman protocol, by fixing two members' published strings as their public keys, the Joux protocol implies a mini broadcast cryptosystem without a trusted dealer in which only these two members can decrypt the messages. Nevertheless, the Joux protocol is limited to three parties. To date, it remains unknown whether it can be extended to more than three parties without introducing additional rounds.

For more than three parties, Boneh and Silverberg proposed a one-round $(n+1)$-party protocol [6] but their protocol relies on n-linear pairings which by itself is also an open problem and no construction has been found so far. Assuming that there exist n-linear pairing maps, their protocol implies a broadcast cryptosystem [4,14] for n users by fixing n users' published strings as their public keys. Similarly to the Joux protocol, the Boneh-Silverberg protocol is limited to $n+1$ parties. It seems difficult to extend it to more than $n+1$ parties without additional rounds even if the existence of efficient n-linear pairings is assumed.

Burmester and Desmedt [11] extended the Diffie-Hellman protocol to n parties. Their protocol requires two rounds and is the most efficient existing GKA protocol in round efficiency without constraints on n. Some papers (*e.g.*, [7, 22]) can achieve authenticated GKA in one-round. But these GKA protocols are based on public key encryption and differ from the above ones in that they are *not from scratch*. The protocol in [7] is PKI-based and only one party uses its public key for encryption purpose. Since setting up a PKI may be regarded as a one-round protocol, such protocols can fairly be compared with a two-round protocol from scratch. It is an open question whether our one-round protocol can be used to build a two-round authenticated GKA from scratch.

However, as remarked by Joux [19], in some cases the two rounds in key agreement protocols can be somewhat cumbersome, and a single pass would be much more preferable. For instance, exchanging an email message key among a

group of users with a two-round key agreement protocol would require all of them to be connected concurrently. Another scenario is a group of friends wishing to share their private files via the insecure Internet; doing so with a two-round key agreement protocol would require all of them to be online at the same time. In practice, it is difficult for a group of distributed users to be online concurrently (especially if they live in different time zones). In these scenarios, the bandwidth is not a problem but the round efficiency is critical. In addition, the bandwidth overhead can be mitigated by the hardware development but the round overhead can only be addressed by more efficient protocols.

Unauthenticated GKA protocols can only be secure against passive attackers. Hence, it is necessary to improve these basic protocols to meet active security for practical applications. There are lots of studies following this research line. In [1] Bellare *et al.* showed a compiler which converts unauthenticated protocols into authenticated ones in the two-party setting; extending [1] to the group setting is possible but inefficient. In [20], Katz and Yung proposed an aggregatable compiler which transforms any secure GKA protocol against passive attackers into a secure authenticated GKA protocol against active attackers. The compiler preserves forward security of the original protocol. In [21], Kudla and Paterson considered modular proof techniques for authenticated key agreement protocols which are not designed in a modular way but which one nevertheless wishes to prove secure. Different key agreement protocols secure against active attackers may be constructed from authentication techniques and a library of basic protocols including our protocols in this paper.

1.3 Motivating Applications

By way of motivation, we illustrate some interesting applications and advantages of the new notion of ASGKA as well as our one-round instantiation.

Application scenarios. Our ASGKA protocol suits broadcast applications to which regular broadcast schemes and GKA protocols are difficult to apply. We have in mind applications where it is hard to find a trusted party to serve as a dealer in a regular broadcast scheme, and it is inconvenient to require all the parties to stay online concurrently to implement a (two-round) regular GKA protocol. Such applications include broadcast to *ad hoc* groups, off-line file sharing via internet, secure group chat, group purchase of encrypted content with identity privacy (*i.e.*, where the seller cannot identify the members of a group purchase) and so on. These applications deal with not very large self-organized groups which live a period of time after being established. Hence, our ASGKA fills the application gap left by regular GKA or broadcast schemes.

Comparison with the trivial solution. As mentioned in Section 1.1, the trivial solution suffers from linear complexity in the ciphertext size, keys storage requirement and encryption overhead. In our scheme, the ciphertext, the negotiated public key and all decryption keys are of constant size. Although in our proposal each member's published string is linear in the group size in negotiation, no one needs to keep any bit of them after executing the protocol. Hence, the storage requirement is small in our scheme. The encryption operation

in our scheme is also efficient, *i.e.*, three exponentiations. Due to the heavy communication overhead in key establishment, our scheme does not improve on the trivial solution for one-time group applications[1]. However, in practice, once a group is established, it is likely to live for a period of time, as discussed above, in which case the communication overhead incurred by our protocol can be amortized over the group lifetime.

Public key based broadcast without a dealer. For our ASGKA proposal, if each member is allowed to register to a certificate authority its published string as its public key, then anyone knowing the public keys of all members can send encrypted messages to the group and only the group members can decrypt the message. Here, the ciphertexs and the secret key of each member are constant and independent of the group scale. This implies that our ASGKA protocol can be used as an efficient public key based dealer-free broadcast scheme which, to the best of our knowledge, is not found in the public literature. However the existing broadcast schemes in the literature do require such a privileged dealer and confidential channels from the dealer to each subscribers before implementing the broadcast system, which is undesirable in some applications.

Key self-confirmation. After a GKA protocol is executed, the agreed keys may become invalid for various reasons, for instance, inside attackers or communication errors. Some proposals (e.g., [10, 20]) suggest a simple method to complete the key confirmation. They assume that the group members have agreed on a public message in advance. After running the basic protocols, each member encrypts this message with its derived session key and broadcasts the ciphertext. Then the members can verify whether they share the same session key (the existing GKAs are symmetric) by checking the consistency of the ciphertexts. This method may not work if there exist inside attackers (a major goal of the key confirmation is to prevent inside attacks). Such attackers can invalidate a correct session key by simply sending a random string in the last round. This flaw can be fixed by letting the group members prove that they have behaved honestly, but such fixes seem expensive in practice. For our asymmetric GKA protocol, the key confirmation is simple and requires no additional rounds if the protocol has been correctly executed. Group members can choose a random message and encrypt it using their derived encryption keys. Then they can decrypt using their derived decryption keys. If the decryption outputs a correct plaintext, the corresponding member concludes that the ASGKA protocol has been correctly run; otherwise, the member raises a public alarm. Hence, if an ASGKA protocol has been correctly executed, each member can verify this fact locally without communicating with others and no additional rounds are required.

Identification of disruptive principals in GKA protocols. In existing GKA protocols, it is not a trivial job to identify malicious principals aiming to disrupt the protocol even if some of these protocols are authenticated and proven secure against active attackers. Indeed, authentication in a GKA protocol only prevents such attacks by outside adversaries. A disruptive principal can just

[1] Those applications are about fully dynamic group broadcast without a dealer and could be interesting for future research.

broadcast some authenticated random strings during the protocol execution to paralyze the protocol. To thwart such attackers, group members may be required to prove that they have behaved correctly in a zero-knowledge manner. This modification can work (to identify the disruptive principals) but it introduces a substantial cost. In our ASGKA scheme, since the transcripts from participants are indeed signatures under their respective temporary public keys, anyone who did not honestly behave can be identified and expelled if any group member raises an alarm after key confirmation.

1.4 Paper Organization

In Section 2, we revisit the notion of group key agreement. Section 3 presents a generic ASGKA construction from ASBBs with key homomorphic property and aggregatability. An ASBB scheme is efficiently realized in Section 4 and the one-round ASGKA protocols naturally follow from the generic formula. Section 5 is a conclusion.

2 Group Key Agreement Revisited

In the conventional GKA definition, a group of members interact to establish a common secret key within an open network. One may notice that a major goal of GKAs is to establish a confidential broadcast channel among the group. In such a broadcast context from GKA protocols, there is no need for a dealer to distribute decryption keys, but only group members can broadcast to other group members[2]. However, in practice the senders can be potentially anyone, as the case of asymmetric encryption. Hence, we are motivated to find alternatives to achieve the final goal of conventional GKAs. For instance, assume that there exists a public key based broadcast cryptosystem in which the single public key corresponds to numerous secret decryption keys; if the group members can jointly generate such a cryptosystem and share the public key while only the group members can extract a secret key during the joint computation, then any message encrypted under this public key can only be decrypted by the group members and a confidential channel is built among them.

2.1 Protocol Variables and Partner Relationship

The following revisited GKA definition derives from the widely-accepted ones due to [7,8,9,10,20]. Similarly to [20], we assume for simplicity a fixed, polynomial-size set $\mathbb{P} = \{\mathcal{U}_1, \cdots, \mathcal{U}_\ell\}$ of potential players. Any subset of the potential players may decide at any point to establish a confidential channel among them, but we assume that these subsets are the same during a run of the protocol and concentrate on static groups.

[2] In some applications, this feature can be an advantage, *e.g.*, intra-group authentications. In our ASGKA protocol, intra-group authentication can be easily realized since the underlying primitive can be used as a signature scheme.

Whenever group membership changes, a new group $\mathbb{P}_v = \{\mathcal{U}_1, \cdots, \mathcal{U}_n\}$ is formed and its members can establish a confidential channel through an instance performing a group key agreement protocol Σ: the index v increases whenever the membership changes and $\mathbb{P}_0$ denotes the initial group. $\Pi_{\mathcal{U}_i}^{\imath_i}$ denotes an instance $\imath_i$ of a group member $\mathcal{U}_i$. An instance $\Pi_{\mathcal{U}_i}^{\imath_i}$ has a unique session identifier $\mathtt{Sid}_{\mathcal{U}_i}^{\imath_i}$ and a partner identifier $\mathtt{Pid}_{\mathcal{U}_i}^{\imath_i}$. After the GKA protocol Σ has been terminated successfully, $\Pi_{\mathcal{U}_i}^{\imath_i}$ has a unique decryption key identifier $\mathtt{Dkid}_{\mathcal{U}_i}^{\imath_i}$ corresponding to a decryption key $dk_{\mathcal{U}_i}^{\imath_i}$, and a unique encryption key identifier $\mathtt{Ekid}_{\mathcal{U}_i}^{\imath_i}$ corresponding to an encryption key $ek_{\mathcal{U}_i}^{\imath_i}$, and a freshness identifier $\mathtt{Fid}_{\mathcal{U}_i}^{\imath_i}$ representing whether $dk_{\mathcal{U}_i}^{\imath_i}$ is compromised. If not, $\mathtt{Fid}_{\mathcal{U}_i}^{\imath_i} = 1$; else $\mathtt{Fid}_{\mathcal{U}_i}^{\imath_i} = 0$. Finally, the partner identifier $\mathtt{Pid}_{\mathcal{U}_i}^{\imath_i}$ corresponds to a set of group members $\mathbb{P}_{\mathcal{U}_i}^{\imath_i} = \mathbb{P}_v \setminus \{\mathcal{U}_i\}$.

Definition 1. (Successful termination of GKA.) *We say that a GKA protocol Σ has been successfully terminated in the instance $\Pi_{\mathcal{U}_i}^{\imath_i}$ if for $1 \leq k \neq i \leq n$, (1) each $\mathcal{U}_k$ of $\mathbb{P}_{\mathcal{U}_i}^{\imath_i}$ has instance $\Pi_{\mathcal{U}_k}^{\imath_k}$ containing $\{\mathtt{Sid}_{\mathcal{U}_k}^{\imath_k}, \mathtt{Pid}_{\mathcal{U}_k}^{\imath_k}, \mathtt{Dkid}_{\mathcal{U}_k}^{\imath_k}, \mathtt{Ekid}_{\mathcal{U}_k}^{\imath_k}\}$; (2) $\mathtt{Sid}_{\mathcal{U}_k}^{\imath_k} = \mathtt{Sid}_{\mathcal{U}_i}^{\imath_i}$; (3) $\mathbb{P}_{\mathcal{U}_k}^{\imath_k} = \mathbb{P}_v \setminus \{\mathcal{U}_k\}$. In this case, we state that the instances $\Pi_{\mathcal{U}_i}^{\imath_i}$ and $\Pi_{\mathcal{U}_k}^{\imath_k}$ are partnered.*

2.2 Adversarial Model

In the real world, a protocol determines how principals behave in response to signals from their environment. In the model, these signals are sent by the adversary $\mathcal{A}$. For simplicity, only passive adversaries are considered in the definitions. A passive adversary is assumed to merely eavesdrop all communication in the network. An adversary $\mathcal{A}$'s interaction with the principals in the network (more specifically, with the various instances) is modeled by the following oracles:

- $\mathtt{Parameter}(1^\lambda)$: On $\mathcal{A}$'s query λ, respond with common parameters denoted by π, including two polynomial time algorithms $\mathcal{E}(\cdot,\cdot)$ and $\mathcal{D}(\cdot,\cdot)$.
- $\mathtt{Setup}(\mathbb{P}_0)$: On $\mathcal{A}$'s query $\mathbb{P}_0$, start the protocol Σ and output the initial group $\mathbb{P}_0 = \{\mathcal{U}_1, \cdots, \mathcal{U}_\ell\}$. For $1 \leq k \leq \ell$, initialize $\mathtt{Sid}_{\mathcal{U}_k}^{\imath_k} \leftarrow 0, \mathtt{Pid}_{\mathcal{U}_k}^{\imath_k} \leftarrow \varnothing, \mathtt{Dkid}_{\mathcal{U}_k}^{\imath_k} \leftarrow NULL, \mathtt{Ekid}_{\mathcal{U}_k}^{\imath_k} \leftarrow NULL, \mathtt{Fid}_{\mathcal{U}_k}^{\imath_k} \leftarrow 1, S \leftarrow 0$.
- $\mathtt{Execute}(\mathcal{U}_1, \cdots, \mathcal{U}_n)$: Execute the protocol between unused instances of players $\{\mathcal{U}_1, \cdots, \mathcal{U}_n\} = \mathbb{P}_v \subseteq \mathbb{P}_0$ and output the transcript of the execution. Here, v changes whenever the group changes and hence is the group sequence number. The number of group members and their identities are chosen by the adversary. For $1 \leq k \leq n$, update $\mathtt{Sid}_{\mathcal{U}_k}^{\imath_k} \leftarrow \mathtt{Sid}_{\mathcal{U}_k}^{\imath_k} + 1$, $\mathtt{Pid}_{\mathcal{U}_k}^{\imath_k} \leftarrow \mathbb{P}_v \setminus \{\mathcal{U}_k\}$, $\mathtt{Dkid}_{\mathcal{U}_k}^{\imath_k} \leftarrow dk_{\mathcal{U}_k}^{\imath_k}$, $\mathtt{Ekid}_{\mathcal{U}_k}^{\imath_k} \leftarrow ek_{\mathcal{U}_k}^{\imath_k}$, $S \leftarrow S + 1$. S is the session sequence number to record the running times of $\mathtt{Execute}$.
- $\mathtt{Ek\text{-}Reveal}(\Pi_{\mathcal{U}_i}^{\imath_i})$: Output $\mathtt{Ekid}_{\mathcal{U}_i}^{\imath_i}$.
- $\mathtt{Dk\text{-}Reveal}(\Pi_{\mathcal{U}_i}^{\imath_i})$: Output $\mathtt{Dkid}_{\mathcal{U}_i}^{\imath_i}$. Update $\mathtt{Fid}_{\mathcal{U}_i}^{\imath_i} \leftarrow 0$. We allow the encryption key to be different from the decryption key and hence the $\mathtt{Ek\text{-}Reveal}$ oracle and the $\mathtt{Dk\text{-}Reveal}$ oracle are distinguished.
- $\mathtt{Test}(\Pi_{\mathcal{U}_i}^{\imath_i}, m_0, m_1)$: This query is used to define the advantage of an adversary $\mathcal{A}$. $\mathcal{A}$ executes this query on a *fresh instance* $\Pi_{\mathcal{U}_i}^{\imath_i}$ (see Definition 4 below) at any time, but only once (other queries have no restriction). When $\mathcal{A}$ asks this query, it receives a challenge ciphertext $c^* = \mathcal{E}(m_\rho, ek_{\mathcal{U}_i}^{\imath_i})$, where ρ is the result of a coin flip. Finally, $\mathcal{A}$ outputs a bit ρ'.

2.3 Properties of GKA

A major goal of GKA protocols is to establish a confidential channel for group members. We use this goal to define the *correctness* of a GKA protocol while use the approaches to the goal to classify GKA protocols.

Definition 2. (Correctness.) *We say a GKA protocol Σ is correct if, whenever it has been successfully terminated, for any instance $\Pi_{\mathcal{U}_i}^{\imath_i}$ and any of its partners $\Pi_{\mathcal{U}_k}^{\imath_k}$ and any message m in the message space of $\mathcal{E}(\cdot,\cdot)$, it holds that $\mathcal{D}(\mathcal{E}(m, ek_{\mathcal{U}_i}^{\imath_i}), dk_{\mathcal{U}_k}^{\imath_k}) = m$ and $\mathcal{D}(\mathcal{E}(m, ek_{\mathcal{U}_k}^{\imath_k}), dk_{\mathcal{U}_i}^{\imath_i}) = m$.*

Definition 3. (Asymmetric Group Key Agreement.) *A GKA protocol Σ is said to be symmetric if after being successfully terminated, it holds that $dk_{\mathcal{U}_i}^{\imath_i} = dk_{\mathcal{U}_k}^{\imath_k} = sk$. Else, Σ is said to be an asymmetric group key agreement protocol.*

A GKA protocol is nontrivial if $|\mathcal{E}(m, ek_{\mathcal{U}_i}^{\imath_i})|$ and $|\mathcal{E}(m, ek_{\mathcal{U}_k}^{\imath_k})|$ are independent of n, where $|s|$ denotes the binary length of string s. This restraint rules out trivial protocols where each member just publishes a public key and keeps its secret key for an agreed public key cryptosystem. The conventional GKA protocols are all symmetric. An ASGKA protocol allows different members to have different decryption keys. However, it does not state whether the inside members can use their secret decryption keys to (collusively) compute a new (equivalent) decryption key. This issue of traitor traceability is beyond our scope.

We first define the notion of *freshness* as the precondition to define the secrecy of GKA protocols, in which the corruption query `Dk-Reveal` models insider attacks and captures forward security of GKAs.

Definition 4. (Freshness.) *An instance $\Pi_{\mathcal{U}_i}^{\imath_i}$ is fresh if before the adversary answers the `Test` oracle, neither the instance $\Pi_{\mathcal{U}_i}^{\imath_i}$ nor anyone of its partnered instances has received a* `Dk-Reveal` *query.*

Let us proceed to the main security notion of GKAs against passive attackers. In the conventional GKA definitions, the group members finally share a secret key. Accordingly, the security is defined by the indistinguishability of the shared key from a random string. In our definition we allow the group members to have different decryption keys. Hence, we define the security of GKAs by the security of the final confidential channel, *i.e.*, the indistinguishability of messages transferred via this channel.

Definition 5. (Secrecy of GKA.) *Let Σ be a GKA protocol and $\mathcal{A}$ be a passive adversary against Σ. When $\mathcal{A}$ asks a query $\texttt{Test}(\Pi_{\mathcal{U}_i}^{\imath_i}, m_0, m_1)$ to a fresh instance $\Pi_{\mathcal{U}_i}^{\imath_i}$ in Σ, $\mathcal{A}$ receives a challenge ciphertext $c^* = \mathcal{E}(m_\rho, ek_{\mathcal{U}_i}^{\imath_i})$, where ρ is the result of a coin flip. Finally, $\mathcal{A}$ outputs a bit ρ'. The advantage of $\mathcal{A}$ in the above secrecy game is defined as $Adv_{\mathcal{A},\Sigma}^{GKA} = |\Pr[\rho' = \rho] - \frac{1}{2}|$. Σ is said to be secure for static groups if $\mathcal{A}$ is allowed to query all the oracles and Adv_{Σ}^{GKA} is negligible.*

3 Generic Construction of One-round ASGKA

In this section, we propose a secure generic construction of one-round ASGKA protocols for static groups. To achieve this goal, we present a primitive referred to as *aggregatable signature-based broadcast* (ASBB). It has the *key-homomorphic property* and *aggregatability* which are crucial for our generic one-round ASGKA protocol.

3.1 Aggregatable Signature-based Broadcast

An ASBB scheme can be used simultaneously as a signature scheme and a broadcast scheme. It exploits the duality between decryption keys and signatures which has been noticed in ID-based encryption [3] and certificate-based encryption [17], but the encryption in ASBB does not take input the receivers' identities.

The security of an ASBB scheme incorporates the standard notion of security for a signature scheme, *i.e.*, existential unforgeability under a chosen message attack (EUF-CMA) [18] and the security as an encryption scheme.

Definition 6. (Aggregatable signature-based broadcast.) *An ASBB scheme consists of six polynomial-time algorithms:*

- $\pi \leftarrow \texttt{ParaGen}(1^\lambda)$*: On input a security parameter* λ*, output the public parameters* π*.*
- $(pk, sk) \leftarrow \texttt{KeyGen}(\pi)$*: On input* π*, output a public/secret key pair* (pk, sk)*.*
- $\sigma \leftarrow \texttt{Sign}(pk, sk, s)$*: On input the key pair* (pk, sk) *and any string* s*, output a signature* σ*.*
- $0/1 \leftarrow \texttt{Verify}(pk, s, \sigma)$*: On input the public* pk *and a signature* σ *of string* s*, output 0 or 1.*
- $c \leftarrow \texttt{Encrypt}(pk, m)$*: On input a public key* pk *and a plaintext* m*, output a ciphertext* c*.*
- $m \leftarrow \texttt{Decrypt}(pk, s, \sigma, c)$*: On input the public key* pk*, any valid string-signature* (s, σ) *and a ciphertext* c*, output the plaintext* m*.*

Given the system parameters π*, a public key* pk *and a challenge ciphertext* $c = \texttt{Encrypt}(pk, m_\rho)$ *for* m_0 *and* m_1 *chosen by an attacker* $\mathcal{A}$*, where* ρ *is the challenger's random coin flip, the attacker wins if it outputs a guess bit* $\rho' = \rho$*. An ASBB scheme is said to be semantically indistinguishable against chosen plaintext attacks (Ind-CPA) if, for any polynomial-time attacker* $\mathcal{A}$*,* $|\Pr[\rho = \rho'] - \frac{1}{2}|$ *is negligible in* λ*. An ASBB scheme is said to be EUF-CMA-Ind-CPA secure if the underlying signature is EUF-CMA secure and the encryption is Ind-CPA secure.*

An ASBB scheme has a key-homomorphic property, as mentioned in Section 1.1. This means that, given two signatures on the same message under two public keys, one can efficiently produce a signature of the same message under a new public key derived from the original two public keys.

Definition 7. (Key homomorphism.) *An ASBB scheme is said to be key homomorphic if, for any two public/secret key pairs* $(pk_1, sk_1), (pk_2, sk_2) \leftarrow \texttt{KeyGen}(\pi)$ *and any message string* s, $\sigma_1 = \texttt{Sign}(pk_1, sk_1, s)$, $\sigma_2 = \texttt{Sign}(pk_2, sk_2, s)$, *it holds that (1)* $\texttt{Verify}(pk_1 \otimes pk_2, s, \sigma_1 \odot \sigma_2) = 1$ *and (2)* $\texttt{Decrypt}(pk_1 \otimes pk_2, s, \sigma_1 \odot \sigma_2, c) = m$ *for any plaintext* m *such that* $c \leftarrow \texttt{Encrypt}(pk_1 \otimes pk_2, m)$, *where* $\otimes : \Gamma \times \Gamma \rightarrow \Gamma$ *and* $\odot : \Omega \times \Omega \rightarrow \Omega$ *are two efficient operations in the public key space* Γ *and the signature space* Ω, *respectively.*

For key homomorphic ASBB schemes sharing system parameters, the combination of signatures of the same string under different public keys is also a valid signature of this string under the combination of given public keys. Note that this property does not contradict the standard EUF-CMA security of signatures, since in a EUF-CMA game, the public key is provided by the challenger while the public key derived from combination is generated by the attacker in the key-homomorphic notion.

Let us consider this problem further. Since the combination of given public keys yields the public key of a new ASBB instance, we naturally question the security of the resulting ASBB instance such as the EUF-CMA security, Ind-CPA security or other properties that are potentially useful. For our focus of one-round GKAs, we study the Ind-CPA security of the resulting system and introduce another important property, *i.e.*, aggregatability, in an ASBB scheme.

Definition 8. (Aggregatability.) *The aggregatability of an ASBB scheme is defined by the following game between an adversary* $\mathcal{A}$ *and a challenger* $\mathcal{CH}$*:*

- **Setup:** $\mathcal{A}$ *initializes the game with an integer* n. $\mathcal{CH}$ *replies with* $(\pi, pk_1, \cdots, pk_n)$ *which are the system parameters and* n *independent public keys of the key-homomorphic ASBB scheme.*
- **Education:** *For* $1 \leq j \neq i \leq n$, *the adversary* $\mathcal{A}$ *can adaptively choose any string* $s_j \in \{0,1\}^*$ *to query* $\mathcal{CH}$ *for a valid signature* $\sigma_i(s_j)$ *under* pk_i. s_j *is determined by the attacker but the limit is that* $s_i \neq s_j$ *if* $i \neq j$. *In other words, the attacker can freely decide* n *messages* $s_j (1 \leq j \leq n)$ *but it cannot query* s_j *to* PK_j, *although the queries to* PK_i *are allowable.*
- **Challenge:** $\mathcal{CH}$ *and* $\mathcal{A}$ *run a standard Ind-CPA game under the combined public key* $pk = pk_1 \otimes \cdots \otimes pk_n$. $\mathcal{A}$ *wins if* $\mathcal{A}$ *outputs a correct guess bit. Denote* $\mathcal{A}$*'s advantage by* $Adv_{\mathcal{A}} = |\Pr[win] - \frac{1}{2}|$.

An ASBB scheme is said to be (τ, ε, n)*-aggregatable against adaptively chosen message attacks if no* τ*-time algorithm* $\mathcal{A}$ *has advantage* $Adv_{\mathcal{A}} \geq \varepsilon$ *in the above aggregatability game. An ASBB scheme is said to be* (τ, ε, n)*-aggregatable against non-adaptively chosen message attacks if the adversary is required to provide* s_j *for* $j = 1, \cdots, n$ *after the* Setup *stage and no* τ*-time algorithm* $\mathcal{A}$ *has advantage* $Adv_{\mathcal{A}} \geq \varepsilon$ *in the corresponding non-adaptive aggregatability game.*

The aggregatability against non-adaptively chosen message attacks is sufficient for our purpose of one-round GKAs. Note that, in the above aggregatability definition, the EUF-CMA security is implicitly used. This is because if the

underlying ASBB is not EUF-CMA secure, the attacker may successfully forge signatures of some message s' under all the public keys after the Education stage. Due to the key-homomorphic property, the attacker can obtain a signature σ' of s' under pk and σ' is also a valid decryption key. As a result, the attacker can decrypt normally and the aggregatability cannot hold.

3.2 A Generic Construction of One-round ASGKA Protocols

Based on ASBBs, we present a generic construction of ASGKA protocols for static groups. The construction is illustrated in Matrix (1), where $\Downarrow/\downarrow$, $\prec$: and TPK represent public/private computation, broadcast operation and temporary public key, respectively.

$$\begin{pmatrix} & & \mathcal{U}_1\texttt{knows} & \mathcal{U}_2\texttt{knows} & \mathcal{U}_3\texttt{knows} & \cdots & \mathcal{U}_n\texttt{knows} & \texttt{TPK} \\ \mathcal{U}_1 & \prec: & \bigcirc & \sigma_1(ID_2) & \sigma_1(ID_3) & \cdots & \sigma_1(ID_n) & pk_1 \\ \mathcal{U}_2 & \prec: & \sigma_2(ID_1) & \bigcirc & \sigma_2(ID_3) & \cdots & \sigma_2(ID_n) & pk_2 \\ \mathcal{U}_3 & \prec: & \sigma_3(ID_1) & \sigma_3(ID_2) & \bigcirc & \cdots & \sigma_3(ID_n) & pk_3 \\ \vdots & & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathcal{U}_n & \prec: & \sigma_n(ID_1) & \sigma_n(ID_2) & \sigma_n(ID_3) & \cdots & \bigcirc & pk_n \\ & & \downarrow & \downarrow & \downarrow & \cdots & \downarrow & \Downarrow \\ \texttt{example:} & & dk_1 & dk_2 & dk_3 & \cdots & dk_n & pk \end{pmatrix} \tag{1}$$

Protocol I: One-round ASGKA Protocol

- `Public Parameter Generation.` The public parameters are the description π of an ASBB scheme.
- `Group Setup.` Decide the group of the players $\mathbb{P} = \{\mathcal{U}_1, \cdots, \mathcal{U}_n\}$. Let $ID_1, \cdots, ID_n$ be the respective identities.
- `Group Key Agreement.` $\mathcal{U}_i$ randomly generates a public key pk_i and broadcasts the corresponding row in Matrix (1):

  $$\sigma_i(ID_1), \cdots, \sigma_i(ID_{i-1}), \sigma_i(ID_{i+1}), \cdots, \sigma_i(ID_n), pk_i.$$

  Here, $\sigma_i(ID_j)$ is a signature of ID_j corresponding to the public key pk_i. $\bigcirc$ *represents that* $\sigma_i(ID_i)$ *is not published for* $1 \leq i \leq n$.
- `Group Encryption Key Derivation.` The group encryption key is:

  $$pk = \bigotimes_{j=1}^{n} pk_j.$$

– `Decryption Key Derivation.` Player $\mathcal{U}_i$ can calculate its secret decryption key dk_i from the corresponding column of Matrix (1):

$$dk_i = \sigma_i(ID_i) \overset{n, j\neq i}{\underset{j=1}{\odot}} \sigma_j(s_i) = \overset{n}{\underset{j=1}{\odot}} \sigma_j(ID_i).$$

Note that an attacker cannot compute dk_i since $\sigma_i(ID_i)$ is unpublished. The last row is the output of a successful run of an ASGKA instance.

– `Encryption/Decryption.` Since π is an ASBB scheme, dk_i is a signature of s_i under the new public key pk. Hence, dk_i is also a decryption key corresponding to the new public key pk and the `Encryption/Decryption` algorithms can be trivially realized in π.

We give the proof of the following theorem in [23].

Theorem 1. *The proposed generic one-round ASGKA protocol for static groups is secure if the underlying key homomorphic ASBB is EUF-CMA-Ind-CPA secure and aggregatable against non-adaptive chosen message attacks.*

Although we only consider the CPA security in our ASGKA protocol, by noting that the resulting output of the protocol can be viewed as a public key based broadcast system, it is possible to achieve security against chosen ciphertext attacks (CCA). In fact, some generic approaches that convert a CPA secure encryption scheme into a CCA secure one can apply to our scheme, similarly to the Fujisaki-Okamoto conversion [15]. Since our focus is one-round ASGKAs with basic security, we will not explore such improvements here.

4 The Proposal

From the generic construction, to realize one-round ASGKA protocols, one needs only to implement a secure ASBB scheme. We construct an ASBB scheme secure in the random oracle model using available bilinear pairing techniques.

4.1 An Efficient ASBB Scheme

The scheme is realized in bilinear pairing groups [5,16]. Let `PairGen` be an algorithm that, on input a security parameter 1^λ, outputs a tuple $\Upsilon = (p, \mathbb{G}, \mathbb{G}_T, e)$, where $\mathbb{G}$ and $\mathbb{G}_T$ have the same prime order p, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is an efficient non-degenerate bilinear map such that $e(g, g) \neq 1$ for any generator g of $\mathbb{G}$, and for all $u, v \in \mathbb{Z}$, it holds that $e(g^u, g^v) = e(g, g)^{uv}$.

– **Public parameters:** Let $\Upsilon = (p, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathtt{PairGen}(1^\lambda)$, $\mathbb{G} = \langle g \rangle$. Let $H : \{0,1\}^* \to \mathbb{G}$ be a cryptographic hash function. The system parameters are $\pi = (\Upsilon, g, H)$.

– **Public/secret keys:** Select at random $r \in \mathbb{Z}_p^*, X \in \mathbb{G} \setminus \{1\}$. Compute

$$R = g^{-r}, A = e(X, g).$$

The public key is $pk = (R, A)$ and the secret key is $sk = (r, X)$.

– **Sign:** The signature of any string $s \in \{0,1\}^*$ under the public key pk is

$$\sigma = XH(s)^r.$$

– **Verify:** Given a message-signature pair (s, σ), the verification equation is

$$e(\sigma, g)e(H(s), R) = A.$$

If the equation holds, output 1 to represent that purported signature is valid. Else output 0 and reject the purported signature.

– **Encryption:** For a plaintext $m \in \mathbb{G}_T$, randomly select $t \in \mathbb{Z}_p^*$ and compute

$$c_1 = g^t, c_2 = R^t, c_3 = mA^t.$$

– **Decryption:** After receiving a ciphertext (c_1, c_2, c_3), anyone with a valid message-signature pair (s, σ) can extract

$$m = \frac{c_3}{e(\sigma, c_1)e(H(s), c_2)}.$$

The correctness of the proposed ASBB scheme follows from a direct verification. Define $\otimes$ by $(R_1, A_1) \otimes (R_2, A_2) = (R_1R_2, A_1A_2)$ and $\odot$ by $\sigma_1 \odot \sigma_2 = \sigma_1\sigma_2$. For security, we have the following claims in which Claim 2 follows from the definition of $\otimes$ and $\odot$, and the security proof of Claim 3 can be found in the full version of the paper [23].

Theorem 2. *Let $\mathbb{G}$ be a bilinear group of prime order p. For any positive integers n, the following claims hold. (1) The proposed ASBB scheme is (τ', ϵ, n)-aggregatable against non-adaptive chosen message attacks in the random oracle model assuming the decision (τ, ϵ, n)-BDHE assumption holds in $\mathbb{G}$, where $\tau' = \tau + \mathcal{O}((q_H + n^2)\tau_{Exp})$. (2) The proposed ASBB scheme is key-homomorphic. (3) The proposed ASBB scheme is EUF-CMA-Ind-CPA secure in the random oracle model under the Computational Diffie-Hellman (CDH) and Decisional Bilinear Diffie-Hellman (DBDH) assumptions.*

Proof. The aggregatability relies on the decision n-BDHE assumption which is shown to be sound by Boneh et al. [2] in the generic group model. The decision n-BDHE assumption in $\mathbb{G}$ is as follows.

Definition 9. *Let $\mathbb{G}$ be bilinear group of prime order p as defined in Section 4.1 and g, h two independent generators of $\mathbb{G}$. Denote $\overrightarrow{y}_{g,\alpha,n} = (g_1, \cdots, g_n, g_{n+2}, \cdots, g_{2n}) \in \mathbb{G}^{2n-1}$, where $g_i = g^{\alpha^i}$ for some unknown $\alpha \in \mathbb{Z}_p^*$. An algorithm $\mathcal{B}$ that outputs $b \in \{0,1\}$ has advantage ε in solving the decision n-BDHE problem if*

$$|\Pr[\mathcal{B}(g, h, \overrightarrow{y}_{g,\alpha,n}, e(g_{n+1}, h)) = 0] - \Pr[\mathcal{B}(g, h, \overrightarrow{y}_{g,\alpha,n}, Z) = 0)]| \geq \varepsilon$$

where the probability is over the random choice of g, h in $\mathbb{G}$, the random choice $\alpha \in \mathbb{Z}_p^*$, the random choice of $Z \in \mathbb{G}_T$, and the random bits consumed by $\mathcal{B}$. We say that the decision (τ, ε, n)-BDHE assumption holds in $\mathbb{G}$ is if no τ-time algorithm has advantage at least ε in solving the decision (τ, ε, n)-BDHE problem.

Based on the decision BDHE assumption, we prove the critical aggregatability of our ASBB scheme. We construct an algorithm $\mathcal{B}$ that uses a decision BDHE challenge to simulate all the requested service for an aggregatability attacker $\mathcal{A}$. Then $\mathcal{B}$ uses the answer bit from $\mathcal{A}$ to solve the the decision BDHE assumption. We illustrate that $\mathcal{B}$ has the same advantage as $\mathcal{A}$ in the same time complexity except for an additive factor, i.e., *the reduction is tight*.

Suppose that $\mathcal{A}$ is a τ-time adversary $\mathcal{A}$ breaking the aggregatability with probability larger than ϵ for a system parameterized with a given n. We build an algorithm $\mathcal{B}$ with advantage ϵ in solving the decision n-BDHE problem in $\mathbb{G}$. $\mathcal{B}$ takes as input a random decision n-BDHE challenge $(g, h, \overrightarrow{y}_{g,\alpha,n}, Z)$, where $\overrightarrow{y}_{g,\alpha,n} = (g_1, \cdots, g_n, g_{n+2}, \cdots, g_{2n})$ and Z is either $e(g_{n+1}, h)$ or a random element of $\mathbb{G}_T$ (recall that $g_i = g^{(\alpha^i)}$). $\mathcal{B}$ proceeds as follows.

Preparation for simulation. For $j = 1, \cdots, n$, $\mathcal{B}$ randomly selects $v_i \in \mathbb{Z}_p$ and computes $h_j = g_j g^{v_j}$. $\mathcal{B}$ randomly selects $i^* \in \{1, \cdots, n\}, a_i, r_i \in \mathbb{Z}_p^*$. Let $S_{i^*} = \{1, \cdots, i^* - 1, i^* + 1, \cdots, n\}$. Compute

$$R_{i^*} = g^{r_{i^*}}(\prod_{k \in S_{i^*}} g_{n+1-k}), \sigma_{i^*,j} = g^{a_{i^*}} g_j^{-r_{i^*}} (\prod_{k \in S_{i^*}}^{k \neq j} g_{n+1-k+j}^{-1}) R_{i^*}^{-v_j} (j \neq i^*).$$

For $i \neq i^*$, compute

$$R_i = g^{r_i} g_{n+1-i}^{-1}, \quad \sigma_{i,j} = g^{a_i} g_j^{-r_i} g_{n+1-i+j} R_i^{-v_j} \text{ for } j \neq i.$$

Noting that $g_i^{(\alpha^j)} = g^{(\alpha^{i+j})} = g_{i+j}$ for any i, j, for $j \neq i^*$, we have that

$$\begin{aligned}
&e(\sigma_{i^*,j}, g) e(h_j, R_{i^*}) = e(g^{a_{i^*}} g_j^{-r_{i^*}} (\prod_{k \in S_{i^*}}^{k \neq j} g_{n+1-k+j}^{-1}) R_{i^*}^{-v_j}, g) e(g_j g^{v_j}, R_{i^*}) \\
&= e(g^{a_{i^*}} g_j^{-r_{i^*}} (\prod_{k \in S_{i^*}}^{k \neq j} g_{n+1-k+j}^{-1}), g) e(g_j, R_{i^*}) \\
&= e(g^{a_{i^*}} g_j^{-r_{i^*}} (\prod_{k \in S_{i^*}}^{k \neq j} g_{n+1-k+j}^{-1}), g) e(g_j, g^{r_{i^*}} (\prod_{k \in S_{i^*}} g_{n+1-k})) \\
&= e(g^{a_{i^*}} g_j^{-r_{i^*}} (\prod_{k \in S_{i^*}}^{k \neq j} g_{n+1-k+j}^{-1}), g) e(g, g_j^{r_{i^*}} (\prod_{k \in S_{i^*}} g_{n+1-k+j})) \\
&= e(g^{a_{i^*}}, g) e(g, g_{n+1}) = e(g, g)^{a_{i^*}} e(g, g)^{\alpha^{n+1}} \stackrel{\texttt{def}}{=} A_{i^*}.
\end{aligned}$$

For $j \neq i (i \neq i^*)$, it holds that

$$\begin{aligned}
&e(\sigma_{i,j}, g) e(h_j, R_i) = e(g^{a_i} g_j^{-r_i} g_{n+1-i+j} R_i^{-v_j}, g) e(g_j g^{v_j}, R_i) \\
&= e(g^{a_i} g_j^{-r_i} g_{n+1-i+j}, g) e(g_j, R_i) = e(g^{a_i} g_j^{-r_i} g_{n+1-i+j}, g) e(g_j, g^{r_i} g_{n+1-i}^{-1}) \\
&= e(g^{a_i}, g) e(g_j^{-r_i} g_{n+1-i+j}, g) e(g, g_j^{r_i} g_{n+1-i+j}^{-1}) = e(g^{a_i}, g) \\
&= e(g, g)^{a_i} \stackrel{\texttt{def}}{=} A_i.
\end{aligned}$$

Hence, for all $j \neq i (i \in \{1, \cdots, n\})$, we have that

$$e(\sigma_{i,j}, g) e(h_j, R_i) = A_i. \tag{2}$$

After the above preparations, $\mathcal{B}$ runs the aggregatability game with attacker $\mathcal{A}$. During the game, $\mathcal{B}$ will give $\mathcal{A}$ the public system parameters including public keys and $\mathcal{A}$ can ask signatures from $\mathcal{B}$ according to Definition 8. We model the hash function as a random oracle maintained by $\mathcal{B}$ and the $\mathcal{A}$ can ask hash outputs with its chosen queries.

Setup simulation. $\mathcal{B}$ needs to generate n public keys $\{pk_1, \cdots, pk_n\}$. $\mathcal{B}$ sets $pk_i = (R_i, A_i)$ and forwards them to $\mathcal{A}$. Note that r_i, a_i are uniformly distributed in $\mathbb{Z}_p^*$. Hence the simulated public keys have an identical distribution as in the

real world and the simulation is perfect. After the Setup stage, $\mathcal{A}$ has to submit its *distinct* queries $s_j \in \{0,1\}^*(1 \leq j \leq n)$ in the Education stage to $\mathcal{B}$.

Random oracle simulation. After Setup stage, $\mathcal{A}$ can query the random oracle at any point. Assume that $\mathcal{A}$ queries the random oracle at most q_H times. For each time, $\mathcal{A}$ queries $\mathcal{B}$ with (s'_j, j) for the hash output of s'_j, where $s'_j \in \{0,1\}^*$ and j is a positive integer. Assume that s'_j is fresh (If s'_j has been queried, $\mathcal{B}$ just returns the previous reply for consistence). To simulate the random oracle, $\mathcal{B}$ works as follows. If $1 \leq j \leq n$ and $s'_j = s_j$, $\mathcal{B}$ sets $H(s'_j) := h_j$; else if $1 \leq j \leq n$ but $s'_j \neq s_j$ or $n < j \leq q_H$, $\mathcal{B}$ randomly selects $h_j \in \mathbb{G}$ and sets $H(s'_j) := h_j$. $\mathcal{B}$ responds with h_j to the query (s'_j, j) from $\mathcal{A}$. Clearly, the simulation of the random oracle is perfect.

Education simulation. For $1 \leq j \leq n$, $\mathcal{B}$ needs to generate signatures $\sigma_i(s_j)$ such that $\texttt{verify}(\sigma_i(s_j), s_j, pk_i) = 1(i \neq j)$. After the above preparation, $\mathcal{B}$ can easily simulate education: When $\mathcal{A}$ requests the signature on s_j under the public key pk_i, $\mathcal{B}$ responds with $\sigma_i(s_j) = \sigma_{i,j}$. From Equation (2), the simulated signatures are well formed and the simulation is also perfect.

Challenge simulation. $\mathcal{B}$ and $\mathcal{A}$ run a standard Ind-CPA game under the the aggregated public encryption key. Denote that $a = a_1 + \cdots + a_n, r = r_1 + \cdots + r_n$. Note that $S_{i^*} = \{1, \cdots, n\} \setminus \{i^*\}$. It follows that the aggregated public encryption key is (R, A) where

$R = R_{i^*} \prod_{k \in S_{i^*}} R_k = g^{r_{i^*}} (\prod_{k \in S_{i^*}} g_{n+1-k}) \prod_{k \in S_{i^*}} g^{r_k} g_{n+1-k}^{-1}$
$= g^{r_{i^*}} \prod_{k \in S_{i^*}} g^{r_k} = g^{\sum_{k=1}^{n} r_k} = g^r,$
$A = A_{i^*} \prod_{k \in S_{i^*}} A_k = e(g,g)^{a_{i^*}} e(g,g)^{\alpha^{n+1}} \prod_{k \in S_{i^*}} e(g,g)^{a_k}$
$= e(g,g)^{\alpha^{n+1}} \prod_{k=1}^{n} e(g,g)^{a_k} = e(g,g)^{\alpha^{n+1}+a}.$

For the aggregated public key (R, A), the decryption keys corresponding to each group member are not revealed. Although neither $\mathcal{B}$ nor $\mathcal{A}$ know the stars $\star$ satisfying $e(\star, g)e(H(s_i), R) = A$, $\mathcal{B}$ knows $r, a \in \mathbb{Z}_p^*$ such that $R = g^r, A = e(g,g)^{\alpha^{n+1}+a}$, where α is unknown. Hence, $\mathcal{B}$ can challenge $\mathcal{A}$ as follows.

When receiving the plaintexts $m_0, m_1 \in \mathbb{G}_T$ chosen by $\mathcal{A}$, $\mathcal{B}$ randomly selects a bit $b \in \{0,1\}$ and computes the challenge ciphertext (c_1^*, c_2^*, c_3^*) where $c_1^* = h, c_2^* = h^r, c_3^* = m_b Z e(g,h)^a$. $\mathcal{B}$ claims that $Z = e(g_{n+1}, h)$ to answer the decision n-BDHE challenge if and only if $\mathcal{A}$'s guess bit $b' = b$.

Success probability: Since g, h are generators, we let $h = g^t$ for some unknown $t \in \mathbb{Z}_p^*$. Hence,

$R^t = (g^r)^t = (g^t)^r = h^r,$
$A^t = (g^r)^t = e(g,g)^{(\alpha^{n+1}+a)t} = e(g, g^t)^{(\alpha^{n+1}+a)}$
$= e(g^{\alpha^{n+1}}, h) e(g,h)^a = e(g_{n+1}, h) e(g,h)^a.$

Therefore, if and only if $Z = e(g_{n+1}, h)$, $(c_1^*, c_2^*, c_3^*) = (g^t, R^t, m_b A^t)$ and (c_1^*, c_2^*, c_3^*) is well formed (*i.e.*, it is a valid ciphertext of m_b under the public key (R, A)). Hence, $\mathcal{B}$ has the same advantage to solve the decision n-BDHE challenge as that of $\mathcal{B}$ breaking the aggregatability of the ASBB scheme.

Time-complexity: In the simulation, $\mathcal{B}$'s overhead is dominated by computing $(\sigma_{i,j}, R_i, A_i)$ for $j \neq i$ and h_j. Computing h_j requires q_H exponentiations in $\mathbb{G}$ (the overhead to sample a random element in $\mathbb{G}$ is about one exponenti-

ation). Computing $\sigma_{i,j}$ requires $\mathcal{O}(n^2)$ exponentiations in $\mathbb{G}$. Note that $\mathcal{B}$ can compute A_i by an exponentiation in $\mathbb{G}_T$ rather than a pairing computation. Computing R_i, A_i requires $\mathcal{O}(n)$ exponentiations in $\mathbb{G}$ and $\mathbb{G}_T$, respectively. Let $\tau_{\mathtt{Exp}}$ denote the time complexity to compute one exponentiation without differentiation of exponentiations in different groups. Hence, the time complexity of $\mathcal{B}$ is $\tau' = \tau + \mathcal{O}((q_H + n^2)\tau_{\mathtt{Exp}})$. □

4.2 Concrete One-round ASGKA Protocol

Following the generic construction, it is easy to realize a one-round ASGKA protocol using the instantiated ASBB scheme. Interestingly, we can remove the hash requirements by observing the randomness in the setup stage. That is, we bind the i-th user by randomly choosing $h_i \in \mathbb{G}$ rather than setting $h_i = H(ID_i)$, while other parts are realized literally following from the generic construction.

- Public parameters generation. It is the same as the above ASBB scheme.
- Group setup. Decide the group of the players $\mathbb{P} = \{\mathcal{U}_1, \cdots, \mathcal{U}_n\}$. Randomly choose $h_i \in \mathbb{G}$ for $i = 1, \cdots, n$. h_i can map to $\mathcal{U}_i$ in a natural way, e.g., according to the dictionary order in their binary representation.
- Group key agreement. $\mathcal{U}_i$ randomly chooses $X_i \in \mathbb{G}, r_i \in \mathbb{Z}_p^*$ and publishes

$$\{\sigma_{i,j}, R_i, A_i\}_{i \neq j}, \text{ where } \sigma_{i,j} = X_i h_j^{r_i}, R_i = g^{-r_i}, A_i = e(X_i, g).$$

- Group encryption key derivation. The players share the same group encryption key(R, A):

$$R = \prod_{j=1}^{n} R_j = g^{-\Sigma_{j=1}^{n} r_j}, A = \prod_{j=1}^{n} A_j = e(\prod_{j=1}^{n} X_j, g).$$

- Decryption key derivation. Using the private input (X_i, r_i) during the protocol execution phase, player $\mathcal{U}_i$ can calculate its secret decryption key from the public communication:

$$\sigma_i = X_i h_i^{r_i} \prod_{j=1}^{n, j \neq i} \sigma_{j,i} = \prod_{j=1}^{n} X_j h_i^{r_j} = (\prod_{j=1}^{n} X_j) h_i^{\Sigma_{j=1}^{n} r_j}.$$

- Encryption. For a plaintext $m \in \mathbb{G}_T$, anyone who knows the public parameters and the group encryption key can output the ciphertext $c = (c_1, c_2, c_3)$, where $t \leftarrow \mathbb{Z}_p, c_1 = g^t, c_2 = R^t, c_3 = mA^t$.
- Decryption. Since $e(\sigma_i, g)e(h_i, R) = A$, each player $\mathcal{U}_i$ can decrypt

$$m = \frac{c_3}{e(\sigma_i, c_1)e(h_i, c_2)}.$$

Corollary 1. *The above n-party ASGKA protocol is secure against passive attackers in the standard model under the decision n-BDHE assumption.*

Proof. The proof is a simple combination of Theorems 1 and 2, but since at the group setup stage, $\mathcal{B}$ can simulate $h_j = g_j g^{v_j}$ with a randomly chosen value $v_i \in \mathbb{Z}_p^*$, it does not need the help of a random oracle. The detailed proof is omitted to avoid repetition. □

5 Conclusions and Future Work

We reconsidered the definition of group key agreement and presented the notion of asymmetric group key agreement. Based on a new notion of aggregatable signature-based broadcast, we presented a generic construction of one-round ASGKA protocols, which can also be used as a broadcast scheme but does not need a trusted dealer to distribute secret keys. Finally, efficient ASBB and one-round ASGKA schemes were instantiated using available bilinear pairings. The proposed ASGKA protocol is secure under the decision BDHE assumption without using random oracles. It fills the application gaps left by conventional GKA and broadcast systems. ASGKA being a new notion, it opens numerous avenues for future research such as round-efficient ASGKAs against active attackers, ASGKAs with traitor traceability and (conditional) reductions between ASGKA and conventional GKA protocols.

Acknowledgments and Disclaimer

The authors gratefully acknowledge Professor Colin Boyd for helping to prepare the final paper and anonymous reviewers for their helpful comments. This paper is partly supported by the Spanish Government through projects CONSOLIDER INGENIO 2010 CSD2007-00004 "ARES" and TSI2007-65406-C03-01 "E-AEGIS", by the Australian ARC Discovery Grant DP0877123, and by the Chinese NSF projects 60673071 and 60873268. The fifth author is also partially supported as an ICREA-Acadèmia researcher by the Government of Catalonia. The views of those authors with the UNESCO Chair in Data Privacy do not necessarily reflect the position of UNESCO nor commit that organization.

References

1. Bellare, M., Canetti, R., Krawczyk, H.: A Modular Approach to the Design and Analysis of Authentication and Key Exchange. In: STOC'98, pp. 419-428. ACM Press (1998)
2. Boneh, D., Boyen, X., Goh, E.J.: Hierarchical Identity Based Encryption with Constant Size Ciphertext. In: Cramer, R. (Ed.) Eurocrypt'05. LNCS 3494, pp. 440-456. Springer, Heidelberg (2005)
3. Boneh, D., Franklin, M.: Identity Based Encryption from the Weil Pairing. SIAM J. of Computing 32(3): 586-615 (2003)
4. Boneh, D., Hamburg, M.: Generalized Identity Based and Broadcast Encryption Schemes. In: J. Pieprzyk (Ed.) Asiacrypt'08. LNCS 5350, pp. 455-470. Springer, Heidelberg (2008)
5. Boneh, D., Lynn, B., Shacham H.: Short Signatures from the Weil Pairing. In: Boyd, C. (Ed.) Asiacrypt'01. LNCS 2248, pp. 514-532. Springer, Heidelberg (2001)
6. Boneh, D., Silverberg, A.: Applications of Multilinear Forms to Cryptography. Contemporary Mathematics 324: 71-90 (2003)
7. Boyd, C., González-Nieto, J.M.: Round-Optimal Contributory Conference Key Agreement. In: Desmedt, Y. (Ed.) PKC'03. LNCS 2567, pp. 161-174. Springer, Heidelberg (2003)

8. Bresson, E., Chevassut, O., Pointcheval, D.: Provably Authenticated Group Diffie-Hellman Key Exchange – The Dynamic Case. In: Boyd, C. (Ed.) Asiacrypt'01. LNCS 2248, pp. 290-309. Springer, Heidelberg (2001)
9. Bresson, E., Chevassut, O., Pointcheval, D.: Dynamic Group Diffie-Hellman Key Exchange under Standard Assumptions. In: Knudsen L.R. (Ed.) Eurocrypt'02. LNCS 2332, pp. 321-336. Springer, Heidelberg (2002)
10. Bresson, E., Chevassut, O., Pointcheval, D., Quisquater, J.J.: Provably Authenticated Group Diffie-Hellman Key Exchange. In: Samarati, P. (Ed.) ACM CCS'01, pp. 255-264. ACM Press (2001)
11. Burmester, M., Desmedt, Y.: A Secure and Efficient Conference Key Distribution System. In: Santis, A.D. (Ed.) Eurocrypt'94. LNCS 950, pp. 275-286. Springer, Heidelberg (1994)
12. Diffie, W., Hellman, M.: New Directions in Cryptography. IEEE Transactions on Information Theory 22(6): 644-654 (1976)
13. ElGamal, T.: A Public Key Cryptosystem and Signature Scheme Based on Discrete Logarithms. IEEE Transaction on Information Theory 31: 467-472 (1985)
14. Fiat, A., Naor, M.: Broadcast Encryption. In: Stinson, D.R. (Ed.) Crypto'93. LNCS 1773, p. 480-91. Springer, Heidelberg (1993)
15. Fujisaki, E., Okamoto, T.: Secure Integration of Asymmetric and Symmetric Encryption. In: Wiener, M.J. (Ed.) Crypto'99. LNCS 1666, pp. 537-554. Springer, Heidelberg (1999)
16. Galbraith, S.D., Rotger, V.: Easy Decision Diffie-Hellman Groups. Journal of Computation and Mathematics 7: 201-218 (2004)
17. Gentry, C.: Certificate-Based Encryption and the Certificate-Revocation Problem. In: Biham E. (Ed.) Eurocrypt'03. LNCS 2656, pp. 272-291. Springer, Heidelberg (2003)
18. Goldwasser, S., Micali, S., Rivest, R.: A Digital Signature Scheme Secure against Adaptive Chosen-message Attacks. SIAM J. Computing 17(2): 281-308 (1988)
19. Joux, A.: A One Round Protocol for Tripartite Diffie-Hellman. J. of Cryptology 17: 263-276 (2004)
20. Katz, J., Yung, M.: Scalable Protocols for Authenticated Group Key Exchange. In: Boneh D. (Ed.) Crypto'03. LNCS 2729, pp. 110-125. Springer, Heidelberg (2003)
21. Kudla, C., Paterson, K.G.: Modular Security Proofs for Key Agreement Protocols. In: Roy, B.K. (Ed.) Asiacrypt'05. LNCS 3788, pp. 549-565. Springer, Heidelberg (2005)
22. Tzeng, W.G., Tzeng, Z.J.: Round Efficient Conference Key Agreement Protocols with Provable Security. In: Okamoto T. (Ed.) Asiacrypt'00. LNCS 1976, pp. 614-627. Springer, Heidelberg (2000)
23. Wu, Q., Mu, Y., Susilo, W., Qin, B., Domingo-Ferrer J.: Asymmetric Group Key Agreement. The full version available at: http://eprint.iacr.org/ (2009)