1-2019

DABKE: Secure deniable attribute-based key exchange framework

Yangguang TIAN
Singapore Management University, ygtian@smu.edu.sg

Yingjiu LI
Singapore Management University, yjli@smu.edu.sg

Guomin YANG
University of Wollongong

Willy SUSILO
University of Wollongong

Yi MU
Fujian Normal University

See next page for additional authors

DOI: https://doi.org/10.3233/JCS-181201

Published in JOURNAL OF COMPUTER SECURITY
2019, Volume: 27 Issue: 2 Pages: 259-275
https://doi.org/10.3233/JCS-181201.

DABKE: Secure Deniable Attribute-Based Key Exchange Framework

Yangguang Tian [a], Yingjiu Li [a], Guomin Yang [b], Willy Susilo [b], Yi Mu [c,*], Hui Cui [d] and Yinghui Zhang [a]

[a] *School of Information Systems, Singapore Management University, Singapore*
E-mails: ygtian@smu.edu.sg, yjli@smu.edu.sg, yinghuizhang@smu.edu.sg
[b] *School of Computing and Information Technology, University of Wollongong, NSW, Australia*
E-mails: gyang@uow.edu.au, wsusilo@uow.edu.au
[c] *Fujian Provincial Key Laboratory of Network Security and Cryptology, College of Mathematics and Informatics, Fujian Normal University, Fuzhou, China*
E-mail: ymu.ieee@gmail.com
[d] *Computer Science and Software Engineering, RMIT University, Victoria, Australia*
E-mail: hui.cui@rmit.edu.au

Abstract. We introduce the first deniable attribute-based key exchange (DABKE) framework that is resilient to impersonation attacks. We define the formal security models for DABKE framework, and propose a generic compiler that converts any attribute-based key exchanges into deniable ones. We prove that it can achieve session key security and user privacy in the standard model, and strong deniability in the simulation-based paradigm. In particular, the proposed generic compiler ensures: 1) a dishonest user cannot impersonate other user's session participation in conversations since implicit authentication is used among authorized users; 2) an authorized user can plausibly deny his/her participation after secure conversations with others; 3) the strongest form of deniability is achieved using one-round communication between two authorized users.

Keywords: Attribute-based Key Exchange, Impersonation Attacks, Strong Deniability, Generic Compiler

1. Introduction

Authenticated key exchange (AKE) protocols are the core cryptographic primitives for many network security standards such as IPSec/IKE, TLS and SSH. AKE aims to share a secret key among multiple users over an insecure communication channel, where users authenticate each other using their identities or public keys. AKE can further be explored in the attribute-based context [1–3]: attribute-based AKE (AB-AKE), which enables fine-grained access control among authorized users. Specifically, AB-AKE provides authentication to authorized users based on whether their attributes satisfy a policy. The AB-AKE mechanism is significantly useful in many real-world applications, such as distributed collaborative systems [1, 3]. In practice, it is desirable for users to communicate with each other based on their roles or responsibilities instead of identities or public keys.

*Corresponding author. E-mail: ymu.ieee@gmail.com.

AB-AKE has some inherent properties such as anonymity [4] and deniability [5], because different authorized users may satisfy the same policy. The anonymity means that authorized users are anonymous from the viewpoint of communication counterparts. Such property is implicitly held in AB-AKE due to the feature of fine-grained authentication. As for deniability, it allows authorized users to deny their participating in conversations if the security of communications is later compromised. In particular, deniability is the most important property in secure messaging protocols [6, 7] such as Off-the-Record (OTR) Messaging protocol [8] and Signal [9]. Since personal or business communications may be revealed or leaked in practice, deniable secure messaging protocols are becoming more important to these private communications.

The inherent anonymity may bring a security risk to AB-AKE. For example, we consider an ad hoc group of users with their associated attributes are participating in a text messaging. The secure key establishment, as a central building block of most text messaging constructions, is vulnerable to impersonation attacks. Roughly speaking, a dishonest receiver can successfully impersonate another receiver to establish a shared secret key with a text sender for subsequent conversations, if receivers with different attributes satisfy the same policy. As a result, the text sender and the impersonator (dishonest receiver) perform the delivery of messages using the established shared secret key.

One may question the importance of preventing AB-AKE from impersonation attacks. To see why it is essential, consider AB-AKE in a *multi-party* setting where it is possible that a malicious authorized user can successfully impersonate other authorized users. For example, a malicious user Alice attempts to impersonate an honest user Bob to establish a conversion with another user Charlie, but the impersonated user Bob was not actually involved in the conversation with Charlie. Such an attack is possible in a multi-party setting due to the inherent anonymous property of attribute-based systems. Although there are some existing works on AB-AKE [1, 2], they have not addressed the issue of impersonation attacks. We stress that in the "design" of AB-AKE, such impersonation attacks naturally exist because authorized users authenticate each other based on their access policies or attributes sets. On the other hand, the impersonation attacks are indeed not desirable when using AB-AKE protocols in real-world applications such as text messaging.

The impersonation attacks are easy to prevent if non-repudiable digital signature schemes are applied to protocol transcripts. However, this contradicts to inherent deniability of AB-AKE. The main goal of this work is to design deniable key exchanges in attribute-based setting while the impersonation-resistance is held. Furthermore, it is also desirable to establish a shared secret key using minimal communication rounds. In this work, we construct a deniable attribute-based key exchange (DABKE) framework with one-round communication in a two-party setting.

We stress that designing "impersonation resistant", "*strongly* deniable" and "non interactive" DABKE framework is a non-trivial task. Recall that the naive way to prevent impersonation attacks is to use non-repudiable digital signature schemes on protocol transcripts. For example, the digital signature based SIGMA protocol [5] can achieve a *weak* form of deniability, but not the *strong* deniability we desired. Such *weak* form of deniability merely allows an accused sender to deny his/her communication with a specific receiver in conversations, but the accused sender cannot deny his/her signed protocol transcripts. While *strong* deniability allows the accused sender to later *plausibly* deny sending messages and participating in a conversation. Another possible way is to use non-malleable zero-knowledge (ZK) proof on protocol transcripts, such as deniable AKE protocol in [10]. Its *strong* deniability is achieved via a challenge-response mechanism for proving the knowledge (e.g., user's secret key) to its peer in a ZK manner. However, its construction is not one-round.

To achieve our design goal, we rely on the implicit authentication (which is used in classic HMQV protocol [11]). It means that Alice knows that the only other party who could compute the shared secret key as her is Bob (and vice versa). More precisely, if Alice later receives a message together with a MAC tag that verifies with respect to the key generated from the session, then Alice is assured that the MAC tag must have been generated by Bob [12]. The implicit authentication is naturally suitable to achieve our design goal, in the sense that it enables the impersonation-resistance while the desired one-round communication is held in a two-party setting.

In summary, deniable attribute-based key exchange DABKE framework has the following unique features.

- *Generic Construction.* We provide a generic framework that is built on top of any implicitly authenticated key exchanges and (ciphertext-policy based) attribute-based encryption schemes. In addition, the generic framework relies on standard model assumptions;
- *Impersonation Resistance.* Authorized users can securely authenticate each other for private conversations, and prevent impersonation attacks by exploiting implicit authentication;
- *Strong Deniability.* Authorized users can plausibly deny their participations after secure and private conversations, even if the security of conversations is completely compromised later;
- *Round Efficiency.* We achieve the strongest form of deniability with one-round communication between two authorized users.

1.1. Related work

Attribute-based Encryption. Sahai and Waters [13] first proposed the fuzzy identity-based encryption, in which users must match at least a certain threshold of attributes. Later, Goyal et al. [14] proposed two types of attribute-based encryption (ABE): Key-policy ABE and Ciphertext-policy ABE. In particular, Bethencourt et al. [15] proposed the first CP-ABE scheme that allowing the ciphertext policies to be very expressive, but the scheme is secure in the generic group model. Large amount of works were followed this trend, some of them [14, 16] are selectively secure under CPA (chosen plaintext attack), and some of them [17, 18] are adaptively secure under CPA/CCA (chosen ciphertext attack) for general policies.

Key Exchange. Burmester and Desmedt [19] introduced several key exchange protocols in the multi-party setting, including star-based, tree-based, broadcast-based and cyclic-based protocols. Later, a few generic transformations [20, 21] were proposed to convert passive-secure group key exchange protocols into active-secure ones. On the security models for key exchange protocols, Bellare and Rogaway [22] introduced the first complexity-theoretic security model for key exchange under the symmetric-key setting. The model was later extended and enhanced under different settings [23–25]. Canetti and Krawczyk [26] later refined the previous models and proposed a new model, known as the CK model, which is widely used in the analysis of many well-known key exchange protocols. Some variants [11, 27] of CK model have also been proposed to allow the adversary to obtain either long-term secret key or ephemeral secret key of the challenge session. The models in [2, 3] were naturally extended from extended CK (eCK) model [27] in the attribute-based setting, which allow adversary to access the master secret key.

Deniable Key Exchange. Deniable authentication was formally introduced by Dwork et al. [28] using the simulation-based paradigm. It requires that the transmitted messages are authenticated, and the simulator's view can be simulated using adversary's knowledge only. Later, Di Raimondo et al. [5] considered deniability of key exchange protocols, and formally presented two definitions: *strong* deniability

and *partial* deniability. In particular, they used novel techniques to analyze strong deniability of SKEME and partial deniability of SIGMA respectively. More precisely, they prove strong deniability of SKEME based on the plaintext awareness of the underlying encryption scheme (in the standard model), and prove partial deniability of SIGMA based on a special “oracle" since non-repudiable digital signature schemes are used for authentication.

Jiang and Safavi-Naini [29] proposed an efficient key exchange protocol with *full* deniability, which allows anyone to prove to a distinguisher (i.e., judge) that the communication between two participants happened. Their deniable key exchange protocol is proven secure under the Bellare-Rogaway model [22] in the public random oracle (pRO) (pRO is introduced by Pass [30], which is a weaker assumption compared to random oracle model). A similar deniable work was proposed by Yao and Zhao [10]. They introduced the first provably secure internet key exchange protocol that provides strong deniability for protocol participants simultaneously. In particular, their strong deniability analysis relies on the restricted random oracle model (it is analogous to Pass’s non-programmable random oracle pRO[31]) and (concurrent) knowledge of exponent assumption (KEA). Note that these solutions [10, 29, 32, 33] are interactive deniable key exchanges.

The implicitly AKE protocols [11, 34, 35] formed another important research line in the literature. They are not only enjoy high performance, but also ensure strong deniability. Based on the well-known HMQV protocol, Yao and Zhao [34] proposed a family of implicitly AKE protocols. In addition, they introduced a new notion: “honest player deniability" (*HP-deniability*). This assumption is the strongest form of deniability for implicitly AKE protocols. Because anyone (including judge) can produce transcripts and session keys that look valid to a judge. The relationship between HP-deniability and strong deniability [5] is analogues to that between honest-verifier zero-knowledge (HVZK) and standard ZK.

Recently, Schage [35] proposed a novel implicitly AKE protocol (TOPAS in short) with full perfect forward secrecy. Interestingly, TOPAS also enjoys strong deniability defined in [5], such that adversary is actually one of session participants. In addition, Unger and Goldberg [6] proposed a newly deniable authenticated key exchange for secure messaging applications. In particular, they introduced the first non-interactive Spawn* that offers forward secrecy and strong deniability against both (partical) on-line and off-line distinguishers. The security of Spawn* relies on the Generalized Universal Composability (GUC) framework [32] in the standard model. We compare our proposed framework with some typical works in Table 1 to highlight our distinction: it shows that our proposed generic framework has user privacy and strong deniability with one-round communication; the generic framework is proven secure in the standard model.

1.2. Paper Organization

In the next section, we present the formal security models to capture the security requirements (namely, session key security, user privacy and strong deniability) of a secure attribute-based key exchange protocol. We present our generic attribute-based secure key exchange compiler and formally prove its security in section 3. The paper is concluded in section 4.

2. Security Model

In this section, we present the security models for our proposed generic deniable attribute-based secure key exchanges. As mentioned in the introduction, a secure and deniable generic framework should

Table 1
A comparative summary for deniable AKE protocols.

Function/Algorithm	[5][a]	[29]	[36]	[34]	[10]	[35]	[6][b]	Ours
User Privacy[c]	✓	×	×	×	×	×	✓	✓
Standard Model	✓	×	×	×	×	×	✓	✓
Non-interactive[d]	×	×	✓	✓	×	✓	✓	✓
KEA-related[e]	✓	×	×	×	✓	×	×	×
Implicit Auth	N/A[f]	N/A	×	✓	×	✓	N/A	✓
†-Deniability[g]	Strong	Full	Partial[h]	Strong[i]	Strong	Strong	Strong	Strong

[a]The SKEME protocol in [5].
[b]The Spawn* protocol in [6].
[c]We consider the user privacy with respect to non protocol participants in this work.
[d]We denote one-round communication as non-interactive, and multi-round communications as interactive in this work.
[e]KEA-related assumptions are used for the proof of strong deniability.
[f]N/A denotes (e.g., plaintext awareness) public key cryptosystem is used to construct deniable AKE.
[g]† denotes the *level/range* of deniability, and a detailed comparison between various deniability models (including full, strong and partial deniability) can be found in Section 2.5.
[h]It denotes the peer-and-time deniability, which is a stronger assumption than the partial deniability offered by SIGMA [5].
[i]It denotes the HP-deniability, which is a stronger assumption than the strong deniability offered by SKEME [5].

achieve several security goals: session key security, user privacy and strong deniability. Note that we formalize user privacy and strong deniability *separately* because deniability and privacy are two different notions. Below we present corresponding security models to capture these requirements.

States. We define a system user set $\mathcal{U}$ with n users, i.e. $|\mathcal{U}| = n$. We say an instance oracle Π^i_U (e.g., session i of user U) may be *used* or *unused*, and a user U has unlimited number of instances called *oracles*. The oracle is considered as unused if it has never been initialized. Each unused oracle Π^i_U can be initialized with a secret key *sk*. The oracle is initialized as soon as it becomes part of a group. After the initialization the oracle is marked as used and turns into the *stand-by* state where it waits for an invocation to execute a protocol operation. Upon receiving such invocation the oracle Π^i_U learns its partner id pid^i_U and turns into a *processing* state where it sends, receives and processes messages according to the description of the protocol. During that stage, the internal state information $state^i_U$ is maintained by the oracle. The oracle Π^i_U remains in the processing state until it collects enough information to compute the session key SK^i_U. As soon as SK^i_U is computed Π^i_U *accepts* and *terminates* the protocol execution meaning that it would not send or receive further messages. If the protocol execution fails then Π^i_U terminates without having accepted.

Partnering. We denote the i-th session of a user U by Π^i_U, the set of attribute by δ_U, and the access structure by Λ_U. We provide the definition of access structure in Section 3.1. Let the partner identifier pid^i_U include the *identities* of participating users (including U) in the i-th session of user U with the condition that $\forall U_j \in \mathsf{pid}^i_U, \Lambda_{U_j}(\delta_U) = 1$. Note that Λ_{U_j} denotes the access structure specified by one of $n-1$ users (e.g., U_j), and $\Lambda_{U_j}(\delta_U) = 1$ means the attributes set δ_U satisfies the access structure Λ_{U_j}. In other words, pid^i_U is a collection of *recognized* participants by the instance oracle Π^i_U. Note that the *recognized* participants mean that authorized users are communicating with user U at some sessions. We also define sid^i_U as the unique session identifier belonging to user U of session i. Specifically, $\mathsf{sid}^i_U = \{m_j\}^n_{j=1}$, where $m_j \in \{0,1\}^*$ is the message transcript (e.g., nonces) among users in pid^i_U. We say two instance oracles Π^i_U and $\Pi^j_{U'}$ are *partners* if and only if $\mathsf{pid}^i_U = \mathsf{pid}^j_{U'}$ and $\mathsf{sid}^i_U = \mathsf{sid}^j_{U'}$.

2.1. Definitions

A deniable attribute-based key exchange DABKE framework consists of the following algorithms:

- Setup: This algorithm takes the security parameter λ as input, outputs master public/secret key pair (mpk, msk).
- KeyGen: This is an interactive algorithm between a user and central authority (CA). User takes master public key mpk as input, outputs a public/secret key pair (pk, sk). While CA takes the master secret key msk, an attributes set δ and public key pk as input, outputs a decryption key dk. Note that we focus on ciphertext-policy setting of attribute-based encryption in this work.
- KeyExchange: This is an interactive algorithm among authorized users. Each authorized user takes his/her secret keys (sk, dk) and his/her counterpart's public keys pk as input, outputs a shared secret key SK.

2.2. Session Key Security

The security model for session key security is defined via a game between an adversary $\mathcal{A}$ and a simulator (i.e., challenger) $\mathcal{S}$. $\mathcal{A}$ is an active adversary with full control of communication channels among all authorized users.

- Setup: $\mathcal{S}$ first generates master public/secret key pair (mpk, msk) for CA and public/secret key pair $\{pk_i, sk_i\}$ for n users by running the corresponding key generation algorithms, where sk_i denote the secret key of user i. In addition, $\mathcal{S}$ honestly generates user's attributes set $\{\delta_i\}$ and corresponding decryption keys $\{dk_i\}$. $\mathcal{S}$ also tosses a random coin b which will be used later in the game.
- Training: $\mathcal{A}$ can make the following queries in arbitrary sequence to $\mathcal{S}$.
  - ∗ send: If $\mathcal{A}$ issues send query in the form of (U, i, m) to simulate a network message for the i-th session of user U, then $\mathcal{S}$ would simulate the reaction of instance oracle Π^i_U upon receiving message m, and returns to $\mathcal{A}$ the response that Π^i_U would generate; If $\mathcal{A}$ issues send query in the form of (U', '*start*', Λ_U), then $\mathcal{S}$ creates a new instance oracle $\Pi^i_{U'}$ and returns $\mathcal{A}$ with the first protocol message under the access structure Λ_U.
  - ∗ long-term secret key reveal: If $\mathcal{A}$ issues a long-term key reveal (or corrupt, for short) query to user i, then $\mathcal{S}$ will return the long-term secret keys (sk_i, dk_i) to $\mathcal{A}$.
  - ∗ ephemeral secret key reveal: If $\mathcal{A}$ issues an ephemeral secret key reveal query to (possibly unaccepted) instance oracle Π^i_U, then $\mathcal{S}$ will return the ephemeral secret key contained in Π^i_U at the moment when the query is asked.
  - ∗ master secret key reveal: If $\mathcal{A}$ issues a master secret key reveal query to CA, then $\mathcal{S}$ will return the master secret key msk to $\mathcal{A}$.
  - ∗ session key reveal: $\mathcal{A}$ can issue session key reveal query to an accepted instance oracle Π^i_U. If the session is accepted (maybe terminated afterwards) successfully, then $\mathcal{S}$ will return the session key to $\mathcal{A}$; Otherwise, a special symbol '$\perp$' is returned to $\mathcal{A}$.
  - ∗ test: This query can only be made to an successfully accepted *fresh* (as defined below) session i of an user U. If the instance oracle Π^i_U has generated a session key, then $\mathcal{S}$ does the following:
    - ∗ If the coin $b = 1$, $\mathcal{S}$ returns the real session key to $\mathcal{A}$;
    - ∗ Otherwise, a random session key is drawn from the session key space and returns to $\mathcal{A}$.

It is also worth noting that $\mathcal{A}$ can continue to issue other queries after the test query. However, the test session must maintain fresh throughout the entire game. Finally, $\mathcal{A}$ outputs b' as its guess for b. If $b' = b$, then $\mathcal{S}$ outputs 1; Otherwise, $\mathcal{S}$ outputs 0.

Freshness. We say an *accepted* instance oracle Π_U^i with attributes set δ_U and access structure Λ_U is fresh if $\mathcal{A}$ does not perform any of the following actions during the game:

- $\mathcal{A}$ issues session key reveal query to Π_U^i or its accepted partnered instance oracles $\Pi_{U'}^j$ (if the latter exists);
- $\mathcal{A}$ issues both long-term secret key reveal query to user U' s.t. $U' \in \mathsf{pid}_U^i$ and ephemeral secret key reveal query for some instances $\Pi_{U'}^j$ partnered with Π_U^i.
- $\mathcal{A}$ issues long-term secret key reveal query to user U' s.t. $U' \in \mathsf{pid}_U^i$ prior to the acceptance of instance Π_U^i and there exist no instance oracles $\Pi_{U'}^j$ partnered with Π_U^i.
  Note that master secret key reveal query to CA is equivalent to the long-term secret key reveal to all users.

We define the advantage of an adversary $\mathcal{A}$ in the above game as

$$\mathtt{Adv}_{\mathcal{A}}(\lambda) = |\Pr[\mathcal{S} \to 1] - 1/2|. \tag{1}$$

Definition 2.1. *We say a DABKE protocol has session key security if for any probabilistic polynomial-time (PPT) $\mathcal{A}$, $\mathtt{Adv}_{\mathcal{A}}(\lambda)$ is a negligible function of the security parameter λ.*

2.3. User Privacy

Informally, an adversary (not authorized users) attempts to identify the authorized users involved in a DABKE protocol. We then define the formal user privacy game between an adversary $\mathcal{A}$ and a simulator $\mathcal{S}$ as follows. Let $\mathcal{A}$ chooses a challenge access structure Λ^* before the game.

- Setup: $\mathcal{S}$ first generates master public/secret key pair (mpk, msk) for the CA and public/secret key pair $\{pk_i, sk_i\}$ for n users by running the corresponding key generation algorithms, where sk_i denote the long-term secret key of user i. In addition, $\mathcal{S}$ honestly generates user's attribute sets $\{\delta_i\}$ and corresponding decryption keys $\{dk_i\}$. $\mathcal{S}$ also tosses a random coin b which will be used later in the game.
- Training: $\mathcal{A}$ is allowed to issue execute, ephemeral secret key reveal, long-term secret key reveal and and session key reveal queries to $\mathcal{S}$. Let $\mathcal{U}'$ denote the users whose attributes set satisfies $\Lambda^*(\delta) = 1$, and $\mathcal{A}$ is not allowed to issue long-term secret key reveal query to users in $\mathcal{U}'$.
  Note that the execute query [25] can be used by (passive) $\mathcal{A}$ to invoke an honest execution of the protocol and obtain a transcript of exchanged messages.
- Challenge: $\mathcal{A}$ randomly selects *two* users $U_i, U_j \in \mathcal{U}'$ as the challenge candidates. $\mathcal{A}$ then sends the challenge candidates to $\mathcal{S}$. $\mathcal{S}$ simulates U_b^* by either $U_b^* = U_i$ if $b = 1$ or $U_b^* = U_j$ if $b = 0$, and generates the protocol transcript using the access structure Λ^*.
  Let the rest users $\{U_r\}$ in $\mathcal{U}'$ interact with the challenge user U_b^*, and $\mathcal{A}$ can access all the communication transcripts among them.

$$\{U_r\}_{r \neq i,j} \leftrightarrow U_b^* = \begin{cases} U_i & b = 1 \\ U_j & b = 0 \end{cases}$$

- Guess: $\mathcal{A}$ outputs b' as its guess for b. If $b' = b$, then $\mathcal{S}$ outputs 1; Otherwise, $\mathcal{S}$ outputs 0.

We define the advantage of $\mathcal{A}$ in the above game as

$$\mathtt{Adv}_{\mathcal{A}}(\lambda) = |\Pr[\mathcal{S} \to 1] - 1/2|.$$

Definition 2.2. *We say a DABKE protocol has* user privacy *if for any PPT $\mathcal{A}$, $\mathtt{Adv}_{\mathcal{A}}(\lambda)$ is a* negligible *function of the security parameter λ.*

2.4. Strong Deniability

Informally, an adversary aims to present a "proof" to a third party (judge), claim that any authorized users were participated in some sessions. We formally define the strong deniability model for DABKE protocols as follows.

Definition 2.3. *Let Σ be an attribute-based key exchange protocol defined by a key generation algorithm* KeyGen *and interactive machines Σ_I, Σ_R specifying the role of the initiator and responder respectively. We say that $(\mathsf{KeyGen}, \Sigma_I, \Sigma_R)$ is a concurrently deniable attribute-based key exchange protocol w.r.t the class* Aux *of auxiliary inputs, if for any PPT $\mathcal{A}$ which runs on input decryption keys $\mathsf{dk} = (dk_1, \cdots, dk_n)$, public keys $\mathsf{pk} = (pk_1, \cdots, pk_n)$ and any auxiliary input $aux \in \mathsf{Aux}$, there exists a simulator $\mathcal{S}$ that, running on the same inputs (including random coins) as $\mathcal{A}$, produces a simulated view which is statistically indistinguishable from the real view of $\mathcal{A}$, where* Aux *models extra information that $\mathcal{A}$ may have obtained in other form such as some legal transcripts eavesdropped by $\mathcal{A}$ from the correctly executed protocols. That is, considering the following two probability distributions where $\mathsf{pk} = (pk_1, \cdots, pk_n)$ is the set of public keys of* honest *users whose secrets keys are not compromised:*

$$\mathsf{Real}(\lambda, aux) = [(dk_i, sk_i, pk_i) \leftarrow \mathsf{KeyGen}(\lambda); (aux, \mathsf{dk}, \mathsf{pk}, \mathsf{View}(aux, \mathsf{dk}, \mathsf{pk}))]$$

$$\mathsf{Sim}(\lambda, aux) = [(dk_i, sk_i, pk_i) \leftarrow \mathsf{KeyGen}(\lambda); (aux, \mathsf{dk}, \mathsf{pk}, \ \mathcal{S}\ (aux, \mathsf{dk}, \mathsf{pk}))]$$

then for all PPT distinguishers Dist *and all $aux \in \mathsf{Aux}$, we have*

$$|\Pr_{x \in \mathsf{Real}(\lambda, aux)}[\mathsf{Dist}(x)] = 1 - \Pr_{x \in \mathsf{Sim}(\lambda, aux)}[\mathsf{Dist}(x)] = 1| \leqslant \epsilon(\lambda). \tag{2}$$

where ϵ is a negligible *function of the security parameter λ. In particular, the actions of* off-line *distinguisher* Dist *are described as follows:*

- *The* Dist *is given the full protocol transcripts and accepted session keys in which $\mathcal{A}$ participated.*
- *The* Dist *is allowed to obtain the secret keys of all participants.*

Remark. As for the strong sense of deniability, $\mathcal{A}$ acts as one of protocol participants. The $\mathsf{View}(aux, \mathsf{dk}, \mathsf{pk})$ means that $\mathcal{A}$ (maliciously) performs the Σ protocol runs with other honest users in a real view. While $\mathcal{S}(aux, \mathsf{dk}, \mathsf{pk})$ means that a simulator $\mathcal{S}$, who is running on the same inputs as $\mathcal{A}$ (including $\mathcal{A}$'s secret key and randomness), simulates an indistinguishable view from a real view to judge.

Table 2
Simulated view with respect to †-deniability

Algorithm/Role	Simulator	Adversary
SIGMA [5]: Partial	Initiator (resp., Responder)	Responder (resp., Initiator)
TOPAS [35]: Strong	Initiator (resp., Responder)	Initiator (resp., Responder)
DIKE [10]: “Full"	Neither of Them	Neither of Them
Ours: Strong	Initiator (resp., Responder)	Initiator (resp., Responder)

2.5. Relationship with Various Deniability Models

We evaluate the relationship between the commonly used deniability models and our proposed strong deniability model, and we remove attribute-based context for fair comparison.

Relation to Partial Deniability. The *digital signature* based key exchange protocol (e.g., SIGMA in [5]) is partially deniable if the runs of the honest initiator with a responder are indistinguishable from runs with another responder. In other words, initiator’s transcripts are “peer-independent". The formal partial deniability model is referred to [5], and we do not consider partial deniability and peer-and-time deniability [36] in this work.

Relation to “Full" Deniability. Some works [29, 33, 36] have used the term “full" deniability to describe strong deniability in [5]. The subtle difference between “full" and strong deniability is the actual role of simulator/adversary. More precisely, in the simulation view of a two-party key exchange setting, the simulator is outside the protocol participants for “full" deniability, while the simulator is one of protocol participants for strong deniability. We consider strong deniability [5] in this work.

Relation to On/Off-line Deniability. On-line deniability means that a protocol participant colludes with the judge during protocol execution. The on-line deniability is a stronger assumption than off-line deniability [6, 37]. Specifically, the judge is allowed to interact with the adversary before or during the protocol execution. Since some impossibility results (e.g., in the symmetric-key setting) observed by Dodis et al. [37] for authentication or key exchange protocols, we consider off-line judge only in this work.

A Complete Comparison. We present a complete comparison between few well-known deniability models for authentication/key exchange protocols and our proposed deniability model (see Table. 2), according to the role of adversary/simulator. We assume an initiator and a responder are running an AKE protocol, and the †-deniability is corresponding to the last function in Table 1.

In addition, we formalize an off-line judge [6] in our proposed deniability model, such that the judge is allowed to obtain the secret keys of protocol participants *after* the protocol execution. Note that our proposed deniability model has forward deniability as specified in [38].

3. Our Construction

In this section, we firstly review the building blocks that will be used in the proposed generic attribute-based key exchange compiler, we then present our proposed generic compiler and its security analysis.

3.1. Building Blocks

Access Structure. Let $\{P_1, \cdots, P_n\}$ be a set of parties. A collection $\Lambda \subseteq 2^{\{P_1,\cdots,P_n\}}$ is monotone if $\forall B, C$: if $B \in \Lambda$ and $B \subseteq C$ then $C \in \Lambda$. An access structure (i.e., monotone access structure) is a collection of non-empty subsets of $\{P_1, \cdots, P_n\}$ (i.e., $\Lambda \subseteq 2^{\{P_1,\cdots,P_n\}} \setminus \{\phi\}$). The sets in Λ are called the authorized sets, and the sets not in Λ are called the unauthorized sets.

Access Tree Λ. Let Λ be a tree representing an access structure. Each non-leaf node of the tree represents a threshold gate, described by its children and a threshold value. If num_i is the number of children of a node x and k_x is its threshold value, then $1 \leqslant k_x \leqslant num_x$. If $k_x = 1$, it is an OR gate; If $k_x = num_x$, it is an AND gate. Each leaf node x of the tree is described by an attribute and a threshold value $k_x = 1$. We define the parent of the node x in the tree by $parent(x)$, the attribute associate with the leaf node x in the tree by $att(x)$, the ordering between the children of every node x in the tree by $index(x)$ (numbered from 1 to num).

Satisfying an Access Tree. Let Λ be an access tree with root R. The Λ_x denote the subtree of Λ rooted at the node x (e.g., $\Lambda = \Lambda_R$). If a set of attributes δ satisfies the access tree Λ_x, we denote it as $\Lambda_x(\delta) = 1$. We compute $\Lambda_x(\delta)$ as follows: If x is a leaf node, then $\Lambda_x(\delta)$ returns 1 iff $att(x) \in \delta$; If x is a non-leaf node, evaluate $\Lambda_{x'}(\delta)$ for all children x' of node x. $\Lambda_x(\delta)$ returns 1 iff at least k_x children return 1.

Ciphertext-Policy Attribute-Based Encryption Scheme: It consists of four algorithms [16]: CP-ABE=(Setup, KeyGen, Enc, Dec).

- Setup: The algorithm takes the security parameter λ as input, outputs master public parameters mpk and master secret key msk.
- KeyGen: The algorithm takes the master secret key msk and an attributes set δ as input, outputs a decryption key dk.
- Enc: The probablistic algorithm takes the master public parameters mpk, a message m and an access structure Λ as input, outputs a ciphertext CT. The CT implicitly contains Λ.
- Dec: The deterministic algorithm takes the master public parameters mpk, a ciphertext CT and the decryption key dk as input, outputs the message m if and only if the attributes set δ satisfies the access structure Λ.

Two-Pass Authenticated Key Exchange (TP-AKE) Protocol: It consists of two algorithms: TP-AKE=(KeyGen, KeyExchange).

- KeyGen: Each user i outputs a long-term secret/public key pair (sk_i, pk_i).
- KeyExchange (KE): This is an interactive algorithm among willing users. It actually consists of the following sub-algorithms.
  - KE.Ephemeral: User i outputs an ephemeral secret/public key pair (ek_i, epk_i); Note that ephemeral secret/public key pair may have NAXOS technique for achieving eCK security, and the detailed technical explanations are referred to [27].
  - KE.EX: Users exchange their ephemeral public keys, i.e., epk_i, epk_j; Note that users also exchange their long-term public keys at post-specified peer setting.
  - KE.KDF: User i executes a key derivation function and obtains $K = \mathsf{KDF} : (sk_i, ek_i, pk_j, epk_j)$ $(j \neq i)$. Note that we consider TP-AKE protocol in a two-party setting for simplicity. More generally, we can define $K = \mathsf{KDF} : (sk_i, ek_i, \{pk_j, epk_j\})$ $(j \neq i)$ in a multi-party setting.

Implicit Authentication. Implicit authentication is mainly used to prevent man-in-the-middle attacks for TP-AKE protocols. Specifically, we analyze the *common* features of TP-AKE protocols as follows: 1) The participants exchange ephemeral public keys only; 2) The key derivation function is composed of his/her own ephemeral/secret keys (ek, sk), and the peer's ephemeral/long-term public keys (epk, pk); 3) The exchanged ephemeral public keys are uniformly distributed in the key space.

Relation to Strong Deniability. Two-pass authenticated key exchange protocols with implicit authentication ensure strong deniability [5], such that one of protocol participants is able to deny his/her session participations. Specifically, one of session participants can randomly chooses the session transcripts (i.e., the uniformly distributed ephemeral public keys), and honestly generate the shared key K under the symmetric-key setting. Furthermore, we discover that many existing TP-AKE protocols (e.g., [11, 27, 34, 35, 39–43] and enormous HMQV variants) can achieve strong deniability (Definition 2 in [5]), we then have the following Lemma.

Lemma 3.1. *TP-AKE protocols with implicit authentication have the strong deniability defined in [5].*

Proof. We assume a two-pass key exchange protocol is executed between Alice and Bob where Alice sends ephemeral public key epk_A to Bob, and vice versa for Bob. We assume that both Alice and Bob aim to establish a shared key K in a real view. Note that two simulated values will be presented to judge: the exchanged ephemeral public keys and the resulting shared key. Also note that adversary is one of protocol participants.

The exchanged ephemeral public key pair (epk_A, epk_B) derives from either the secret key pair (ek_A, ek_B) or the function $f(ek_A, sk_A), f(ek_B, sk_B)$ (f denotes a randomized algorithm, such as hash function or pseudorandom function). Simulator $\mathcal{S}$ (i.e., adversary Bob) is simply choosing a random value epk_A from ephemeral public space to simulate transmitted ephemeral public key on behalf of Alice, while the judge cannot *statistically* distinguish it since the real/simulated value is *uniformly* distributed in either the ephemeral public key space or the output of function f (except collisions with a negligible probability). Note that the judge is not allowed to obtain either ephemeral or long-term secret keys of protocol participants (refer to Definition 2 in [5]).

The resulting shared key K can be easily simulated by $\mathcal{S}$. Specifically, $\mathcal{S}$ (i.e., adversary Bob) randomly chooses ephemeral public key epk_B and computes $K = \mathsf{TP-AKE.KE.KDF} : (pk_A, epk_A, sk_B, ek_B)$, where secret keys (sk_B, ek_B) are known to $\mathcal{S}$. □

Pseudo-Random Function (PRF). A function family $\tilde{\mathsf{F}}$ is associated with $\{\mathsf{Seed}_\lambda\}_{\lambda\in\mathbb{N}}$, $\{\mathsf{Dom}_\lambda\}_{\lambda\in\mathbb{N}}$, $\{\mathsf{Rng}_\lambda\}_{\lambda\in\mathbb{N}}$, where $\mathbb{N}$ is the set of natural numbers. Formally, for any randomly chosen $\mathcal{K} \xleftarrow{R} \mathsf{Seed}_\lambda$, $\sigma \xleftarrow{U} \mathcal{K}$, $\mathcal{D} \xleftarrow{R} \mathsf{Dom}_\lambda$, $\mathcal{R} \xleftarrow{R} \mathsf{Rng}_\lambda$, $\mathsf{F}_\sigma^{\lambda,\mathcal{K},\mathcal{D},\mathcal{R}}$ define a function which map an element of $\mathcal{D}$ to an element of $\mathcal{R}$. That is, $\mathsf{F}_\sigma^{\lambda,\mathcal{K},\mathcal{D},\mathcal{R}}(\rho) \in \mathcal{R}$ for any $\rho \in \mathcal{D}$.

Definition 3.1 (PRF [41])**.** *We say that $\tilde{\mathsf{F}}$ is a pseudo-random function (PRF) family if*

$$\{\tilde{\mathsf{F}}_\sigma^{\lambda,\mathcal{K},\mathcal{D},\mathcal{R}}(\rho_i)\} \approx \{\mathsf{RF}(\rho_i)\} \tag{3}$$

for any $\rho_i \in \mathcal{D}$ adaptively chosen by any polynomial time distinguisher, where RF is a truly random function. That is, for any $\rho \in \mathcal{D}$, $\mathsf{RF}(\rho) \xleftarrow{U} \mathcal{R}$.

3.2. Generic Compiler

High-level Description. Two authorized users aim to establish a secure session key using their attributes and public keys, meanwhile a subsequent deniable communication is provided among authorized users. A generic compiler (GC) is used for adding strong deniability and impersonation-resistance to attribute-based key exchanges, and the proposed generic compiler consists of the following building blocks.

- An eCK secure TP-AKE protocol $\mathsf{TP-AKE} = (\mathsf{KeyGen}, \mathsf{KeyExchange})$.
- An ABE scheme $\mathsf{ABE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$.
- A PRF $\tilde{\mathsf{F}}$.

For simplicity, we use initiator U_i and responder U_j to illustrate our generic compiler (see Figure 1 below).

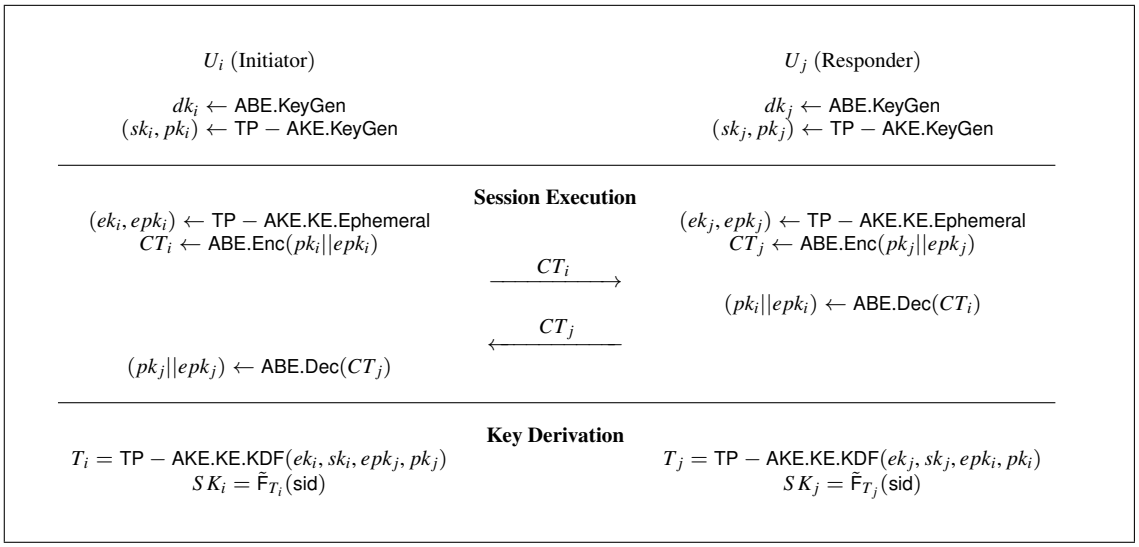

Fig. 1. Generic Compiler. $\mathsf{sid} = (epk_i||epk_j)$

- Setup: CA takes the security parameter λ as input, outputs master public/secret key pair (mpk, msk). Let $\tilde{\mathsf{F}} : \mathcal{K} \times \mathcal{D} \rightarrow \{0,1\}^{l(\lambda)}$ (l is polynomial in the security parameter λ, and $l(\lambda) \in \mathcal{R}$) be a PRF.
- KeyGen: CA runs the $\mathsf{ABE.KeyGen}$ algorithm to obtain decryption key dk_i of user i. In addition, user i runs the $\mathsf{TP\text{-}AKE.KeyGen}$ algorithm to obtain his own long-term secret/public key pair (sk_i, pk_i).
- $\mathsf{KeyExchange}$: User i (i.e., initiator) performs the following steps.

  (1) Run the $\mathsf{TP\text{-}AKE.KE.Ephemeral}$ algorithm to obtain ephemeral secret/public key pair (ek_i, epk_i);
  (2) Generate a ciphertext $CT_i = \mathsf{ABE.Enc}(pk_i||epk_i)$ and send it to user j. Note that at CP-ABE case, $\mathsf{ABE.Enc}$ takes master public key mpk, a message $m_i = (pk_i||epk_i)$ (where epk_i is also used for key exchange KE of $\mathsf{TP\text{-}AKE}$) and an access structure Λ_i as input, outputs a ciphertext CT_i which implicitly contains Λ_i.

  Then user j (i.e., responder) performs the following steps.

  (1) Obtain $(pk_i||epk_i)$ after running $\mathsf{ABE.Dec}$ algorithm on CT_i; Note that (CP-) $\mathsf{ABE.Dec}$ takes master public key mpk, a ciphertext CT_i and the decryption key dk_j as input, outputs the message m_i iff the attributes set δ_j satisfies the access structure Λ_i.

(2) Run the TP-AKE.KE.Ephemeral algorithm to obtain ephemeral secret/public key pair (ek_j, epk_j);
(3) Generate a ciphertext $CT_j = \mathsf{ABE.Enc}(pk_j||epk_j)$ and send CT_j to user i;
(4) Compute the final session key $SK_j = \tilde{\mathsf{F}}_{T_j}(\mathsf{sid})$, where $T_j = \mathsf{TP-AKE.KE.KDF}(ek_j, sk_j, epk_i, pk_i), \mathsf{sid} = (epk_i||epk_j)$.

User i performs the following steps.

(1) Obtain $(pk_j||epk_j)$ after running ABE.Dec algorithm on CT_j;
(2) Compute the final session key $SK_i = \tilde{\mathsf{F}}_{T_i}(\mathsf{sid})$, where $T_i = \mathsf{TP-AKE.KE.KDF}(ek_i, sk_i, epk_j, pk_j), \mathsf{sid} = (epk_i||epk_j)$. Note that the equation $T_i = T_j$ holds due to the correctness of TP-AKE protocol.

Remark. One may argue that the proposed GC is transforming from a TP-AKE protocol to an attribute-based one. Essentially, we aim to prevent impersonation attacks and provide strong deniability for AB-AKE protocols. To achieve these goals, the proposed GC can be construted from either AB-AKE protocol with implicit authentication or TP-AKE protocol with ABE scheme, which in turn provides more flexibility in the sense of generic construction.

3.3. Security Analysis

Theorem 3.2. *The proposed GC achieves session key security (Definition 2.2) if the* TP-AKE *is session key secure and* $\tilde{\mathsf{F}}$ *is a pseudo-random family.*

Proof. We define a sequence of games $\mathbb{G}_i$, $i = 0, \cdots, 4$ and let $\mathtt{Adv}_i^{GC}$ denote the advantage of the adversary in game $\mathbb{G}_i$. Assume that $\mathcal{A}$ activates at most m sessions in each game.

- $\mathbb{G}_0$ This is original game for session key security.
- $\mathbb{G}_1$ This game is identical to game $\mathbb{G}_0$ except that $\mathcal{S}$ will output a random bit if user i (initiator) and user j (responder) accept, but $\mathsf{pid}_i \neq \mathsf{pid}_j, \mathsf{sid}_i \neq \mathsf{sid}_j$. Since n users involved in this game, we have:

$$\left|\mathtt{Adv}_0^{GC} - \mathtt{Adv}_1^{GC}\right| \leqslant n \cdot m^2/2^{\lambda} \tag{4}$$

- $\mathbb{G}_2$: This game is identical to game $\mathbb{G}_1$ except the following difference: $\mathcal{S}$ randomly chooses $g \in [1, m]$ as a guess for the index of the test session. $\mathcal{S}$ will output a random bit if $\mathcal{A}$'s test query does not occur in the g-th session. Therefore we have

$$\mathtt{Adv}_1^{GC} = m \cdot \mathtt{Adv}_2^{GC} \tag{5}$$

- $\mathbb{G}_3$ This game is identical to game $\mathbb{G}_2$ except that in the g-th session, $\mathcal{S}$ replaces the real TP-AKE session key by a random value $R \in \mathcal{K}$. Below we show the difference between $\mathbb{G}_2$ and $\mathbb{G}_3$ is negligible under the assumption that TP-AKE is eCK secure. Note that we choose eCK security of TP-AKE in this game, which is used to match our proposed session key security defined in Section 2.2.
  Let $\mathcal{S}$ denotes an attacker against the TP-AKE protocol, who is given corresponding oracles in the sense of eCK security model, and aims to distinguish a real session key from a random value. $\mathcal{S}$ simulates the game for $\mathcal{A}$ as follows.

* Setup: $\mathcal{S}$ sets up the game for $\mathcal{A}$ by creating n users with the corresponding attributes/identity set $\{\delta_i, ID_i\}$. $\mathcal{S}$ honestly generates master public/secret key pair (mpk, msk) for CA. In addition, $\mathcal{S}$ generates user's decryption keys $\{dk_i\}$ by running $\mathsf{ABE.KeyGen}$ algorithm and public/secret key pairs $\{(pk_i, sk_i)\}$ from his oracle queries (i.e., long-term secret key query). Eventually, $\mathcal{S}$ sends all attribute sets, identities and public keys to $\mathcal{A}$.
* $\mathcal{S}$ answers $\mathcal{A}$'s queries as follows.
  * If $\mathcal{A}$ issues send query in the form of $CT_j = \mathsf{Enc}(pk_j||epk_j)$ to user i, then $\mathcal{S}$ will perform the simulation as follows.
    - Obtain messages pk_j, epk_j using decryption key of user i;
    - Forward epk_j to his challenger and obtain epk_i from his send oracle;
    - Return $CT_i = \mathsf{Enc}(pk_i||epk_i)$ as the response to $\mathcal{A}$;
    - Compute the final session key $SK_i = \tilde{\mathsf{F}}_{T_i}(\mathsf{sid})$, where T_i is obtained from either session key reveal oracle or test oracle w.r.t. g-th session, and $\mathsf{sid} = (pk_i||pk_j||epk_i||epk_j)$.

    Note that if $\mathcal{A}$ issues send query in the form of $(U', start', \Lambda_{U'})$, then $\mathcal{S}$ returns $CT' = \mathsf{Enc}(pk_{U'}||epk')$ (epk' is randomly chosen by $\mathcal{S}$) to $\mathcal{A}$.
  * If $\mathcal{A}$ issues an ephemeral key reveal query to $\mathcal{S}$, then $\mathcal{S}$ makes an ephemeral key reveal query to its own oracle to get ek_i and returns it to $\mathcal{A}$.
  * If $\mathcal{A}$ issues a long-term key reveal query to user i, then $\mathcal{S}$ returns (dk_i, sk_i) to $\mathcal{A}$.
  * If $\mathcal{A}$ issues a master secret key reveal query to $\mathcal{S}$, then $\mathcal{S}$ returns msk to $\mathcal{A}$.
  * $\mathcal{S}$ answers the session key reveal query and test query by using the session key SK_i it has derived during the protocol simulation described above.

  Note that in the test session $\mathcal{S}$ will get either a real session key or a random key from his own test oracle. If it is the real session key, then the simulation is consistent with $\mathbb{G}_2$; Otherwise, the simulation is consistent with $\mathbb{G}_3$. Therefore, if the advantage of $\mathcal{A}$ is significantly different in $\mathbb{G}_2$ and $\mathbb{G}_3$, $\mathcal{S}$ can break the $\mathsf{TP\text{-}AKE}$ protocol. Hence, we have

$$\left|\mathtt{Adv}_2^{GC} - \mathtt{Adv}_3^{GC}\right| \leqslant \mathtt{Adv}_{\mathcal{S}}^{\mathsf{TP-AKE}}(\lambda) \tag{6}$$

- $\mathbb{G}_4$: This game is identical to game $\mathbb{G}_3$ except that in the test session, we replace the PRF function $\tilde{\mathsf{F}}_{T_i}(\mathsf{sid})$ by a random function. We next to showing the difference between $\mathbb{G}_3$ and $\mathbb{G}_4$ is negligible if $\tilde{\mathsf{F}}$ is PRF family.

  Let $\mathcal{S}$ denotes an attacker against the PRF, who is given an oracle either $\tilde{\mathsf{F}}_\sigma$ ($\sigma \xleftarrow{U} \mathcal{K}$) or a random function RF, aims to distinguish them. $\mathcal{S}$ simulates the game for $\mathcal{A}$ as the exact same way as $\mathbb{G}_3$ except the computation of final session key $SK_i = \tilde{\mathsf{F}}_{T_i^*}(\mathsf{sid}^*)$. Specifically, $\mathcal{S}$ forwards sid^* to his challenger and sets the value returned from the oracle as SK_i. Finally, $\mathcal{S}$ outputs whatever $\mathcal{A}$ outputs. Note that in the test session, if the oracle is $\tilde{\mathsf{F}}_\sigma$ (note that the returned value $\tilde{\mathsf{F}}_\sigma(\mathsf{sid}^*)$ from the oracle is indistinguishable from $\tilde{\mathsf{F}}_{T_i^*}(\mathsf{sid}^*)$ because the value $T_i^* \xleftarrow{U} \mathcal{K}$ in $\mathbb{G}_3$ is uniform and independent), then the simulation is consistent with $\mathbb{G}_3$; Otherwise, the simulation is consistent with $\mathbb{G}_4$. Therefore, if the advantage of $\mathcal{A}$ is significantly different in $\mathbb{G}_3$ and $\mathbb{G}_4$, $\mathcal{S}$ can break the PRF family. Hence, we have

$$\left|\mathtt{Adv}_3^{GC} - \mathtt{Adv}_4^{GC}\right| \leqslant \mathtt{Adv}_{\mathcal{S}}^{PRF}(\lambda) \tag{7}$$

It is easy to see that in game $\mathbb{G}_4$, $\mathcal{A}$ has no advantage, i.e.,

$$\mathtt{Adv}_4^{GC} = 0 \tag{8}$$

Combining the above results together, we have

$$\mathtt{Adv}_{\mathcal{A}}^{GC}(\lambda) = n \cdot m^2/2^{\lambda} + m \cdot (\mathtt{Adv}_{\mathcal{S}}^{\mathsf{TP-AKE}}(\lambda) + \mathtt{Adv}_{\mathcal{S}}^{PRF}(\lambda)).$$

□

Remark. The impersonation attacks can be prevented in an implicit manner. If adversary can perform the impersonation attacks with a non-negligible probability, then we can use it to built an efficient attacker to break the session key security of TP-AKE protocol.

Theorem 3.3. *The proposed GC achieves user privacy (Definition 2.3) if the ABE scheme is selective IND-CPA secure.*

Proof. Let $\mathcal{S}$ denote an attacker who is given the KeyGen oracle and challenge access structure Λ^* (CP-ABE case), aims to break the selective IND-CPA (sIND-CPA) security of ABE scheme. $\mathcal{S}$ simulates the game for $\mathcal{A}$ as follows.

- Setup: $\mathcal{S}$ sets up the game for $\mathcal{A}$ by creating n users with the corresponding attributes/identity set $\{\delta_i, ID_i\}$. Then $\mathcal{S}$ honestly generates user's public/secret key pairs $\{(pk_i, sk_i)\}$. Eventually, $\mathcal{S}$ sends all attribute sets, identities and public keys to $\mathcal{A}$.
- Training: $\mathcal{S}$ answers $\mathcal{A}$'s queries as follows.
  - If $\mathcal{A}$ issues an execute query between user i and user j ($\Lambda^*(\delta_i) = 1$ and $\Lambda^*(\delta_j) = 1$), then $\mathcal{S}$ randomly chooses ephemeral secret keys (ek_i, ek_j), and performs the session execution and session key derivation honestly according to the protocol specification.
  - $\mathcal{S}$ answers ephemeral secret key reveal and session key reveal queries using the secret values it has derived during the protocol simulation described above. In addition, $\mathcal{S}$ can honestly answer long-term secret key reveal queries to users $\Lambda^*(\delta_w) \neq 1$ ($w \neq i, j$) using his KeyGen oracle.
- Challenge: After receiving user i and user j ($\Lambda^*(\delta_i) = 1$ and $\Lambda^*(\delta_j) = 1$) from $\mathcal{A}$, $\mathcal{S}$ firstly follows the user privacy game to select U_b^* and construct message $m_0 = (pk_b || epk_b)$. Secondly, $\mathcal{S}$ randomly chooses another user l ($\Lambda^*(\delta_l) = 1$), and generates another message $m_1 = (pk_l || epk_l)$. Thirdly, $\mathcal{S}$ sends challenge messages (m_0, m_1) to his challenger oracle, and obtains challenge ciphertext C^* as the transcript of U_b^*. Note that epk_b, epk_l are randomly chosen by $\mathcal{S}$, and a randomly chosen value R acts as the transcript of user l.

Finally, $\mathcal{S}$ outputs whatever $\mathcal{A}$ outputs. If $\mathcal{A}$ guesses the random bit correctly, then $\mathcal{S}$ can break the selective IND-CPA security of ABE. Hence, we have

$$\mathtt{Adv}_{\mathcal{A}}^{GC}(\lambda) \leqslant \mathtt{Adv}_{\mathcal{S}}^{sIND-CPA}(\lambda).$$

□

Theorem 3.4. *The proposed GC achieves strong deniability defined in Definition 2.4.*

Proof. We assume the GC is executed between Alice and Bob where Alice sends CT_A to Bob, and vice versa for Bob. We assume that both Alice and Bob aim to establish a shared session key SK in a real view. Note that two simulated values will be presented to a judge: the exchanged ciphertexts and the resulting session key. Also note that adversary is one of protocol participants who can decrypt the ciphertext from its counterpart.

Simulator $\mathcal{S}$ (i.e., adversary Bob) simulates Alice's transcript as follows: $\mathcal{S}$ randomly chooses ephemeral public key epk_A as described in Lemma 3.1, and generates ciphertext as $CT_A \leftarrow \mathsf{ABE.Enc}(pk_A||epk_A)$. As for Bob's transcript and resulting session key, $\mathcal{S}$ can generate ciphertext CT_B and session key SK_B by following the GC execution honestly. Specifically, the ciphertext is $CT_B \leftarrow \mathsf{Enc}(pk_B||epk_B)$ and the resulting session key is $SK_B = \tilde{\mathsf{F}}_{T_B}(\mathsf{sid})$, where $T_B = \mathsf{TP-AKE.KE.KDF}(ek_B, sk_B, epk_A, pk_A), \mathsf{sid} = (pk_A||pk_B||epk_A||epk_B)$, and $(ek_B, epk_B) \leftarrow \mathsf{TP-AKE.KE.Ephemeral}$. Note that the secret keys (sk_B, ek_B) are known to $\mathcal{S}$.

We stress that an off-line judge is allowed to obtain long-term secret keys (sk_A, sk_B) rather than ephemeral secret keys, the transcripts simulated by $\mathcal{S}$ are *statistically* indistinguishable from a real view because the randomly chosen value (e.g., epk_A) in simulated transcript is *uniformly* distributed as in a real view. Since $\mathcal{S}$ can simulate the transcripts and resulting session key in the exact same way as a real protocol execution between Alice and Bob, the simulation by $\mathcal{S}$ is perfect to a judge.

□

4. Conclusion

In this paper, we proposed a deniable attribute-based secure key exchange framework based on the attribute-based encryption schemes and implicitly authenticated key exchange protocols. We defined the formal security models to capture the security requirements, including session key security, user privacy and strong deniability. We leave the construction of attribute-based secure key exchange with on-line deniability in the standard model as our future work.

Acknowledgements

The work is supported by the Singapore National Research Foundation under NCR Award Number NRF2014NCR-NCR001-012.

References

[1] M.C. Gorantla, C. Boyd and J.M.G. Nieto, Attribute-Based Authenticated Key Exchange, in: *ACISP*, 2010, pp. 300–317.

[2] K. Yoneyama, Strongly Secure Two-Pass Attribute-Based Authenticated Key Exchange, in: *Pairing-Based Cryptography*, 2010, pp. 147–166.

[3] Y. Tian, G. Yang, Y. Mu, K. Liang and Y. Yu, One-Round Attribute-Based Key Exchange in the Multi-party Setting, in: *Provable Security*, 2016, pp. 227–243.

[4] V. Kolesnikov, H. Krawczyk, Y. Lindell, A. Malozemoff and T. Rabin, Attribute-based key exchange with general policies, in: *ACM, CCS*, 2016, pp. 1451–1463.

[5] M. Di Raimondo, R. Gennaro and H. Krawczyk, Deniable authentication and key exchange, in: *ACM, CCS*, 2006, pp. 400–409.

[6] N. Unger and I. Goldberg, Deniable key exchanges for secure messaging, in: *ACM, CCS*, 2015, pp. 1211–1223.

[7] N. Unger and I. Goldberg, Improved Strongly Deniable Authenticated Key Exchanges for Secure Messaging, *Proceedings on Privacy Enhancing Technologies* **2018**(1) (2018), 21–66.

[8] N. Borisov, I. Goldberg and E. Brewer, Off-the-record communication, or, why not to use PGP, in: *Proceedings of the 2004 ACM workshop on Privacy in the electronic society*, 2004, pp. 77–84.

[9] Open Whisper Systems.

[10] A.C.-C. Yao and Y. Zhao, Privacy-preserving authenticated key-exchange over Internet, *IEEE Transactions on Information Forensics and Security* **9**(1) (2014), 125–140.

[11] H. Krawczyk, HMQV: A High-Performance Secure Diffie-Hellman Protocol, in: *CRYPTO*, 2005, pp. 546–566.

[12] S.D. Galbraith, Authenticated key exchange for SIDH (2018).

[13] A. Sahai and B. Waters, Fuzzy Identity-Based Encryption, in: *Advances in Cryptology - EUROCRYPT*, 2005, pp. 457–473.

[14] V. Goyal, O. Pandey, A. Sahai and B. Waters, Attribute-based encryption for fine-grained access control of encrypted data, in: *ACM, CCS*, 2006, pp. 89–98.

[15] J. Bethencourt, A. Sahai and B. Waters, Ciphertext-Policy Attribute-Based Encryption, in: *2007 IEEE Symposium on Security and Privacy (S&P)*, 2007, pp. 321–334.

[16] B. Waters, Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization, in: *PKC*, 2011, pp. 53–70.

[17] A.B. Lewko, T. Okamoto, A. Sahai, K. Takashima and B. Waters, Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption, in: *EUROCRYPT*, 2010, pp. 62–91.

[18] T. Okamoto and K. Takashima, Fully Secure Functional Encryption with General Relations from the Decisional Linear Assumption, in: *CRYPTO*, 2010, pp. 191–208.

[19] M. Burmester and Y. Desmedt, Efficient and Secure Conference-Key Distribution, in: *Security Protocols, International Workshop, Cambridge, United Kingdom, April 10-12, 1996, Proceedings*, 1996, pp. 119–129.

[20] J. Katz and M. Yung, Scalable Protocols for Authenticated Group Key Exchange, in: *CRYPTO*, 2003, pp. 110–125.

[21] J. Katz and J.S. Shin, Modeling insider attacks on group key-exchange protocols, in: *ACM, CCS*, 2005, pp. 180–189.

[22] M. Bellare and P. Rogaway, Entity Authentication and Key Distribution, in: *CRYPTO*, 1993, pp. 232–249.

[23] M. Bellare and P. Rogaway, Provably secure session key distribution: the three party case, in: *STOC*, 1995, pp. 57–66.

[24] M. Bellare, R. Canetti and H. Krawczyk, A Modular Approach to the Design and Analysis of Authentication and Key Exchange Protocols (Extended Abstract), in: *STOC*, 1998, pp. 419–428.

[25] M. Bellare, D. Pointcheval and P. Rogaway, Authenticated Key Exchange Secure against Dictionary Attacks, in: *EUROCRYPT*, 2000, pp. 139–155.

[26] R. Canetti and H. Krawczyk, Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels, in: *EUROCRYPT*, 2001, pp. 453–474.

[27] B.A. LaMacchia, K.E. Lauter and A. Mityagin, Stronger Security of Authenticated Key Exchange, in: *Provable Security*, 2007, pp. 1–16.

[28] C. Dwork, M. Naor and A. Sahai, Concurrent zero-knowledge, *Journal of the ACM (JACM)* **51**(6) (2004), 851–898.

[29] S. Jiang and R. Safavi-Naini, An efficient deniable key exchange protocol, *Financial Cryptography and Data Security* (2008), 47–52.

[30] R. Pass, On Deniability in the Common Reference String and Random Oracle Model., in: *Crypto*, Vol. 3, 2003, pp. 316–337.

[31] M. Yung and Y. Zhao, Interactive zero-knowledge with restricted random oracles, in: *TCC*, 2006, pp. 21–40.

[32] S. Walfish, Enhanced security models for network protocols, PhD thesis, New York University, 2008.

[33] Y. Tian, Y. Li, Y. Zhang, N. Li, G. Yang and Y. Yu, DSH: Deniable Secret Handshake Framework, in: *ISPEC*, 2018, pp. 341–353.

[34] A.C.-C. Yao and Y. Zhao, OAKE: a new family of implicitly authenticated diffie-hellman protocols, in: *ACM, CCS*, 2013, pp. 1113–1128.

[35] S. Schäge, TOPAS: 2-Pass Key Exchange with Full Perfect Forward Secrecy and Optimal Communication Complexity, in: *ACM, CCS*, 2015, pp. 1224–1235.

[36] C.J.F. Cremers and M. Feltz, One-round Strongly Secure Key Exchange with Perfect Forward Secrecy and Deniability, *IACR Cryptology ePrint Archive* **2011** (2011), 300.

[37] Y. Dodis, J. Katz, A.D. Smith and S. Walfish, Composability and On-Line Deniability of Authentication, in: *TCC*, 2009, pp. 146–162.

[38] M. Di Raimondo and R. Gennaro, New approaches for deniable authentication, in: *ACM, CCS*, 2005, pp. 112–121.

[39] L. Law, A. Menezes, M. Qu, J. Solinas and S. Vanstone, An efficient protocol for authenticated key agreement, *Designs, Codes and Cryptography* **28**(2) (2003), 119–134.

[40] A. Menezes, Another look at HMQV, *Mathematical Cryptology JMC* **1**(1) (2007), 47–64.

[41] T. Okamoto, Authenticated key exchange and key encapsulation in the standard model, *ASIACRYPT* (2007), 474–484.

[42] B. Ustaoglu, Obtaining a secure and efficient key agreement protocol from (H) MQV and NAXOS, *Designs, Codes and Cryptography* **46**(3) (2008), 329–342.

[43] Y. Tian, S. Zhang, G. Yang, Y. Mu and Y. Yu, Privacy-preserving k-time authenticated secret handshakes, in: *ACISP*, 2017, pp. 281–300.