11-2016

One-round attribute-based key exchange in the multi-party setting

Yangguang TIAN

Guomin YANG
Singapore Management University, gmyang@smu.edu.sg

Yi MU

Kaitai LIANG

Yong YU

Citation

TIAN, Yangguang; YANG, Guomin; MU, Yi; LIANG, Kaitai; and YU, Yong. One-round attribute-based key exchange in the multi-party setting. (2016). *Proceedings of the 10th International Conference, Nanjing, China, 2016 November 10–11*. 10005, 227-243.
Available at: https://ink.library.smu.edu.sg/sis_research/7365

One-Round Attribute-Based Key Exchange in the Multi-party Setting

Yangguang Tian[1(✉)], Guomin Yang[1], Yi Mu[1], Kaitai Liang[2], and Yong Yu[3]

[1] School of Computing and Information Technology,
University of Wollongong, Wollongong, NSW 2522, Australia
{ytian,gyang,ymu}@uow.edu.au
[2] School of Computing, Mathematics and Digital Technology,
Manchester Metropolitan University, Manchester M1 5GD, UK
kaitailiang88@gmail.com
[3] School of Computer Science, Shaanxi Normal University, Xi'an 710062, China
yyucd2012@gmail.com

Abstract. Attribute-based authenticated key exchange (AB-AKE) is a useful primitive that allows a group of users to establish a shared secret key and at the same time enables fine-grained access control. A straightforward approach to design an AB-AKE protocol is to extend a key exchange protocol using attribute-based authentication technique. However, insider security is a challenge security issue for AB-AKE in the multi-party setting and cannot be solved using the straightforward approach. In addition, many existing key exchange protocols for the multi-party setting (e.g., the well-known Burmester-Desmedt protocol) require multiple broadcast rounds to complete the protocol. In this paper, we propose a novel one-round attribute-based key exchange (OAKE) protocol in the multi-party setting. We define the formal security models, including session key security and insider security, for OAKE, and prove the security of the proposed protocol under some standard assumptions in the random oracle model.

Keywords: Attribute-based cryptography · One-round key exchange · Multi-party setting · Insider security

1 Introduction

Authenticated key exchange (AKE) protocols are a central building block in many network security standards such as IPSec, TLS/SSL, SSH, and so on. AKE aims to share a common secret key among multiple users over an insecure communication channel, such that the users can authenticate each other by using the respective identities or public keys. AKE has been further explored in the attribute-based context recently [16,26]. Attribute-based AKE (AB-AKE), as a new general form of AKE, enables fine-grained access control between authenticated users. The AB-AKE mechanism is significantly useful in many

L. Chen and J. Han (Eds.): ProvSec 2016, LNCS 10005, pp. 227–243, 2016.
DOI: 10.1007/978-3-319-47422-9_13

real-world applications, such as distributed collaborative systems [16]. In practice, sometimes it is necessary for users to communicate with each other based on their role/responsibility. For instance, an individual user should be allowed to establish a secure communication with another user if and only if the former's role/responsibility can satisfy the latter's expectation.

The anonymity property naturally exists in attribute-based systems since people with different attribute sets may all satisfy an access policy. This brings a security issue for AB-AKE in the multi-party setting where it might be possible that a malicious authorized member can successfully impersonate other authorized members (i.e., insider attacks). Specifically, a malicious user Alice (attacker) attempts to impersonate an honest user Bob to establish a conversion with another user, say Charlie, but the impersonated user Bob was not actually involved in the particular conversion with Charlie. Such an attack is possible due to the inherent anonymous property of attribute-based systems. Therefore, achieving insider security is a non-trivial task for AB-AKE under the multi-party setting. Although there are some existing works on AB-AKE [16,26], they didn't consider the issue of insider attacks in the multi-party setting.

In order to achieve the insider security for AB-AKE in the multi-party setting, in this paper, we propose a novel hybrid signcryption (HSC) scheme to address the issue, where the hybrid signcryption scheme is built on top of a combination of key-policy attribute-based encryption (KP-ABE) [17] and identity-based signature (IBS) [14,19]. In addition to insider security and fine-grained authentication for key exchange, it is also desirable to share a secret key with less communication rounds. Many existing multi-party (or group) AKE protocols, such as the well-known Burmester-Desmedt protocol [12], require mutiple broadcast rounds in order to complete the protocol. In this paper we tackle this problem by making use of the (generic) multilinear maps [10] to establish a session key with only one broadcast round for a group of users.

1.1 This Work

In this paper, we introduce the notion of one-round attribute-based broadcast key exchange in the multi-party setting, allowing all users to agree on a common session key in only one broadcast round. Our contributions are as follows:

1. We present the formal security definitions for OAKE. In particular, we extend the model of [26] to define session key security and propose a new insider security model to capture malicious insider attacks.
2. We introduce a new primitive named hybrid signcryption (HSC), and propose a concrete scheme that is built on top of an identity-based signature scheme and Goyal et al.'s [17] key-policy attribute-based encryption scheme. We also prove that the proposed HSC scheme can achieve existential unforgeability in the random oracle model.
3. We present a one-round AB-AKE protocol in the multi-party setting based on our proposed HSC scheme and the generic multilinear maps [10]. We further prove that the proposed protocol can achieve both session key security and insider security.

1.2 Related Work

Key Exchange. Burmester and Desmedt [12] 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 [8,20, 21] were proposed to convert passive-secure group key exchange protocols into active-secure ones. Bellare and Rogaway [5] introduced the first complexity-theoretic security model for key exchange under the symmetric-key setting. The model was later extended and enhanced under different contexts [3,4,6]. Canetti and Krawczyk [13] 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 [22,23] of CK model were also proposed to allow an adversary to obtain either long term secret key or ephemeral secret key of the challenge session. In [26] an extension of the eCK (extended CK) model proposed in [23] was introduced for the attribute-based setting. It strengthens the session key security model by allowing adversary to gain access to the master secret key.

Signcryption. Zheng [27] introduced the concept of signcryption that provides an efficient way of achieving both message confidentiality and authenticity. The security of the scheme was later proven in [2]. An et al. [1] formally analyzed three generic constructions of signcryption in the public key setting, namely "encrypt then sign" (*EtS*), "sign then encrypt" (*StE*), and "commit then encrypt and sign" (*CtE&S*). Meanwhile, Haber and Pinkas [18] proposed a combined public key scheme under the joint security model, where encryption schemes and signature schemes shared the common public parameters and secret key. Boyen [11] introduced an efficient identity-based signcryption (IBSC) scheme based on the Boneh–Franklin IBE [9] scheme and the Cha–Cheon IBS [14] scheme.

Attribute-based Cryptography. For achieving fine-grained access control over the encrypted data, Sahai and Waters [25] proposed the fuzzy identity-based encryption, in which users must match at least a certain threshold of attributes before data decryption. Later, two types of attribute-based encryption (ABE) systems were proposed: Key-policy ABE [17] and Ciphertext-policy ABE [7]. In KP-ABE, a ciphertext is labeled with an attribute set, while a secret key is associated with the access structure specifying which ciphertext a user is able to decrypt. The roles of attribute set and access structure are swapped in the CP-ABE context. Inspired by the attribute-based cryptography, several attribute-based signcryption schemes [15,24] have been proposed in the literature where both signing and encryption functions are attribute-based. We should note that such kind of attribute-based signcryption schemes are not suitable for our purpose since they cannot address the insider attacks.

2 Security Models

In this section, we present the security models for OAKE. As mentioned in the introduction, a secure OAKE protocol in the multi-party setting should

achieve both session key security and insider security. Below we present the corresponding security models to capture the above requirements. Specifically, the session key security model is a *modified* version of Yoneyama's model [26] which is an extension of eCK model [23] in the attribute-based setting.

States. We define a system user set $\mathcal{U}$ with n users, i.e. $|\mathcal{U}| = n$. We say an oracle Π_U^i may be *used* or *unused*. The oracle is considered as unused if it has never been initialized. Each unused oracle Π_U^i can be initialized with a secret key x. 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_U^i learns its partner id pid_U^i 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 $state_U^i$ is maintained by the oracle. The oracle Π_U^i remains in the *processing* state until it collects enough information to compute the session key k_U^i. As soon as k_U^i is computed Π_U^i *accepts* and *terminates* the protocol execution meaning that it would not send or receive further messages. If the protocol execution fails then Π_U^i terminates without having accepted.

Partnering. We denote the i-th session established by an user U by Π_U^i, the attribute set of the user by δ_U, and the access structure of the user by Λ_U. Let the partner identifier pid_U^i includes the identities of participating users (including U) in the i-th session established by the user U with the condition that $\forall U_j \in \mathsf{pid}_U^i, \Lambda_U(\delta_{U_j}) = 1$, where δ_{U_j} denotes the attribute set of the user U_j, and $\Lambda_U(\delta_{U_j}) = 1$ means that the attribute set δ_{U_j} is satisfied by the access structure Λ_U. In other words, pid_U^i is a collection of *recognized* participants by the instance oracle Π_U^i. We also define sid_U^i as the unique session identifier belonging to the session i established by the user U. Specifically, $\mathsf{sid}_U^i = (\mathsf{pid}_U^i, \{m_j\}_{j=1}^n)$, where $m_j \in \{0,1\}^*$ is the message transcript among users in pid_U^i. We say two instance oracles Π_U^i and $\Pi_{U'}^j$ are *partners* if and only if $\mathsf{sid}_U^i = \mathsf{sid}_{U'}^j$.

2.1 Session Key Security

We define the session key security model for key-policy AB-AKE protocols, in which each user obtains a secret key associating with his/her access structure from the trusted authority (TA), and establishes a session key depending on the partners' attribute sets. The model is defined via a game between a probabilistic polynomial time (PPT) adversary $\mathcal{A}$ and a simulator $\mathcal{S}$. $\mathcal{A}$ is an active attacker with full control of the communication channel among all the users.

- Setup: $\mathcal{S}$ first generates master public/secret key pair $(\mathcal{K}_0, x_0)$ for the TA and long term secret keys $\{x_i\}_{i=1}^n$ for n users by running the corresponding key generation algorithms, where x_i denotes the secret key of user i, such that x_i $(i \neq 0)$ is corresponding to the access structure Λ_i and identity ID_i of user 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 simulator $\mathcal{S}$.

- Establish: $\mathcal{A}$ is allowed to register a user U' with an access structure Λ'. If a party is registered by the $\mathcal{A}$, then we call the user i *dishonest*; Otherwise, it is *honest*.
- Send: If $\mathcal{A}$ issues a 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_U^i upon receiving message m, and returns to $\mathcal{A}$ the response that Π_U^i would generate; If $\mathcal{A}$ issues a send query in the form of $(U,' start', \delta_U)$, then $\mathcal{S}$ creates a new instance oracle Π_U^i and returns to $\mathcal{A}$ the first protocol message under the attribute set δ_U.
- Session key reveal: $\mathcal{A}$ can issue session key reveal query to an accepted instance oracle Π_U^i. If the session is accepted, then $\mathcal{S}$ will return the session key to $\mathcal{A}$; Otherwise, a special symbol '$\perp$' is returned to $\mathcal{A}$.
- Ephemeral secret key reveal: If $\mathcal{A}$ issues an ephemeral secret key reveal query to (possibly unaccepted) instance oracle Π_U^i, then $\mathcal{S}$ will return all ephemeral secret values contained in Π_U^i at the moment the query is asked.
- Long term secret key reveal: If $\mathcal{A}$ issues a long term secret key reveal (or corrupt, for short) query to user i, then $\mathcal{S}$ will return the long term secret key x_i to $\mathcal{A}$.
- Master secret key reveal: If $\mathcal{A}$ issues a master secret key reveal query to TA, then $\mathcal{S}$ will return the master secret key x_0 to $\mathcal{A}$.
- Test: This query can only be made to an accepted and *fresh* (as defined below) session i of a user U. Then $\mathcal{S}$ does the following:
  - If the coin $b = 1$, $\mathcal{S}$ returns the real session key to the adversary;
  - Otherwise, a random session key is drawn from the session key space and returned to the adversary.

  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 the simulator outputs 1; Otherwise, the simulator outputs 0.

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

- $\mathcal{A}$ issues *establish* query, where the new user $U' \in \mathsf{pid}_U^i$;
- $\mathcal{A}$ issues a *session key reveal* query to Π_U^i or its accepted partnered instance oracle $\Pi_{U'}^j$ (if the latter exists);
- $\mathcal{A}$ issues both *long term secret key reveal* query to U' s.t. $U' \in \mathsf{pid}_U^i$ and *ephemeral secret key reveal* query for an instance $\Pi_{U'}^j$ 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 exists no instance oracle $\Pi_{U'}^j$ partnered with Π_U^i.

  Note that the *master key reveal query* is equivalent to the *long term secret key reveal* to all users in $\mathcal{U}$.

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

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

Definition 1. *We say an OAKE protocol has* session key security *if for any probabilistic polynomial-time (PPT) $\mathcal{A}$, $\mathtt{Adv}_{\mathcal{A}}(k)$ is a* negligible *function of the security parameter k.*

2.2 Insider Security

Informally, a PPT adversary $\mathcal{A}$ attempts to impersonate one honest user to communicate with other honest users, whereas the impersonated honest user is not actually involved in that conversion. We define the insider security game between a PPT insider adversary $\mathcal{A}$ and a simulator $\mathcal{S}$ as follows.

- Training: $\mathcal{A}$ is allowed to issue establish, send, ephemeral secret key reveal, at most *n-1* long term secret key reveal, and session key reveal queries to the simulator. Let $\mathcal{U}'$ denotes the set of *uncorrupted* users (the established users are excluded from $\mathcal{U}'$). At the end of training stage, $\mathcal{A}$ outputs (U, U', s), such that $U \in \mathcal{U}'$, i.e., U denotes an *impersonated* but honest user who is *not corrupted*, and $U' \in \mathcal{U}$ can be a *corrupted* user who has a used oracle $\Pi^s_{U'}$. Note that $\mathcal{A}$ is not allowed to issue the master key reveal query, otherwise all users are *corrupted*.
- Attack: $\mathcal{A}$ wins the game if all of the following conditions hold.
  - $\Pi^s_{U'}$ accepted, it implies $\mathsf{sid}^s_{U'}$ exist;
  - $U \in \mathsf{pid}^s_{U'}$, it implies $\Lambda_{U'}(\delta_U) = 1$;
  - $m_U \in \mathsf{sid}^s_{U'}$, *but* there exists *no* Π^i_U which has sent m_U (m_U denotes the message transcript from the user U).

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

$$\mathtt{Adv}_{\mathcal{A}}(k) = \Pr[\mathcal{A}\ wins].$$

Definition 2. *We say a OAKE protocol has* insider security *if for any probabilistic polynomial-time (PPT) $\mathcal{A}$, $\mathtt{Adv}_{\mathcal{A}}(k)$ is an* negligible *function of the security parameter k.*

3 OAKE Protocol

In this section, we firstly review the preliminaries and the building blocks that will be used in the proposed hybrid signcryption scheme and the OAKE protocol, and then introduce our constructions.

3.1 Preliminaries

(Generic) Multilinear Maps [10]**.** It assumes the existence of a group generator g, which takes a security parameter k and the number of levels $\mathcal{K}$ as input, outputs a sequence of groups $(\mathbb{G}_1, \cdots, \mathbb{G}_\mathcal{K})$ with the corresponding canonical generators $(g_1, \cdots, g_\mathcal{K})$, each of them with large prime order q. The multilinear maps $\hat{\mathsf{e}} : \mathbb{G}_i \times \mathbb{G}_j \to \mathbb{G}_{i+j} | i, j \geq 1; i + j \leq \mathcal{K}$ satisfies the following relation:

$$\hat{\mathsf{e}}(g_i^{\alpha_i}, g_j^{\alpha_j}) = g_{i+j}^{\alpha_i \cdot \alpha_j} : \forall \alpha_i, \alpha_j \in_R \mathbb{Z}_q, i + j \leq \mathcal{K}$$

Bilinear Maps. The bilinear maps (i.e., $\mathcal{K} = 2$) $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ has the following properties:

1. Bilinearity: $e(g^{\alpha_i}, g^{\alpha_j}) = e(g, g)^{\alpha_i \cdot \alpha_j} : \forall \alpha_i, \alpha_j \in \mathbb{Z}_q, g \in \mathbb{G}$.
2. Non-degeneracy: $e(g, g) \neq 1$.
3. Computable: There exists an efficient algorithm for computing the bilinear maps.

Note that the maps e is symmetric since $e(g^{\alpha_i}, g^{\alpha_j}) = e(g, g)^{\alpha_i \cdot \alpha_j} = e(g^{\alpha_j}, g^{\alpha_i})$.

$\mathcal{K}$-Multilinear Decisional Diffie-Hellman ($\mathcal{K}$-MDDH) Assumption [10]**:** Given $g_1, g_1^{c_1}, \cdots, g_1^{c_\mathcal{K}}$ where $c_1, \cdots, c_\mathcal{K} \in_R \mathbb{Z}_q$, we define the advantage of the adversary in solving the $\mathcal{K}$-MDDH problem as

$$\mathtt{Adv}_\mathcal{A}(k) = \Pr[b \in \{0, 1\}, \mathcal{A}(g_1^{c_1}, \cdots, g_1^{c_\mathcal{K}}, T_b = g_{\mathcal{K}-1}^{\prod_{i=1}^{\mathcal{K}} c_i}, T_{1-b} \in \mathbb{G}_{\mathcal{K}-1}) = b].$$

The $\mathcal{K}$-MDDH assumption holds if for any PPT $\mathcal{A}$, $\mathtt{Adv}_\mathcal{A}(k)$ is a *negligible* function of the security parameter k.

Computational Diffie-Hellman (CDH) Assumption [22]**:** Given $g, g^a, g^b \in \mathbb{G}$ where $a, b \in_R \mathbb{Z}_q$, we define the advantage of the adversary in solving the CDH problem as

$$\mathtt{Adv}_\mathcal{A}^{CDH}(k) = \Pr[\mathcal{A}(g, g^a, g^b) = g^{ab} \in \mathbb{G}].$$

The CDH assumption holds if for any PPT $\mathcal{A}$, $\mathtt{Adv}_\mathcal{A}(k)$ is a *negligible* function of the security parameter k.

3.2 Building Blocks

Key-Policy Attribute-based Encryption Access Structure [17]**.** 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., $\Lambda \subseteq 2^{\{P_1, \cdots, P_n\}} \backslash \{\phi\}$). The sets in Λ are called the authorized sets, and the sets not in Λ are called the unauthorized sets.

Access Tree Λ [17]**.** 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 $1 \leq k_x \leq num_x$. 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 associates 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 denotes 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.

Key-Policy Attribute-Based Encryption Scheme: It consists of four algorithms [17]: KP-ABE=(Setup, KeyGen, Encrypt, Decrypt).

- Setup: The algorithm takes the security parameter k as input, outputs the master public parameters mpk and the master secret key msk.
- KeyGen: The algorithm takes the master secret key msk and an access structure Λ as input, outputs a secret key sk.
- Encrypt: The algorithm takes the master public parameters mpk, a message M and a set of attributes δ as input, outputs a ciphertext C. The C implicitly contains δ.
- Decrypt: The algorithm takes the master public parameters mpk, a ciphertext C and the secret key sk as input, outputs the message M if and only if the attribute set δ satisfies the access structure Λ.

Identity-Based Signature. An identity-based signature (IBS) scheme [14,19] consists of four algorithms: IBS=(Setup, KeyGen, Sign, Verify).

- Setup: The algorithm takes the security parameter k as input, outputs the master public parameters mpk and the master secret key msk.
- KeyGen: The algorithm takes the master secret key msk and an identity ID as input, outputs a secret signing key sk.
- Sign: The algorithm takes a message M and the signing key sk as input, outputs a signature σ on the message M.
- Verify: The algorithm takes the signature σ, the message M and the identity ID as input, outputs 1 if σ is valid on M, otherwise reject.

Hybrid Signcryption. A hybrid signcryption (HSC) scheme consists of four algorithms: HSC=(Setup, KeyGen, Sigcrypt, Unsigncrypt).

- Setup: The algorithm takes the security parameter k as input, and outputs the master key pair (mpk, msk).

- KeyGen: The algorithm takes an identity ID, an access structure Λ and the master secret key msk as input, and outputs the decryption/signing key pair (dk, sk), where dk is corresponding to Λ and sk is corresponding to ID.
- Signcrypt: The randomized algorithm takes the master public key mpk, a message M, a sender's identity ID, the signing key sk and an attribute set δ as input, and outputs a signcryption CT.
- Unsigncrypt: The deterministic algorithm takes the master public key mpk, a signcryption CT, the decryption key dk as input, and outputs the message M and sender's identity ID if CT is valid, otherwise it outputs reject.

We define an unforgeability game between an insider adversary $\mathcal{A}$ and a simulator $\mathcal{S}$ in the multiple party setting, which proceeds as follows:

- Setup: $\mathcal{S}$ runs $(mpk, msk) \leftarrow \mathsf{Setup}(1^k)$, where k is the security parameter, returns mpk to $\mathcal{A}$.
- Training: $\mathcal{A}$ is allowed to issue Signcrypt, Unsigncrypt and KeyGen queries. Note that $\mathcal{S}$ will return the secret key pair (dk, sk) to $\mathcal{A}$ when issuing the KeyGen query.
- Forgery: $\mathcal{A}$ outputs a signcryption CT^* and an access structure Λ^*.
- Outcome: $\mathcal{A}$ wins if all of the following conditions hold.
  - $\mathsf{Unsigncrypt}(mpk, CT^*, dk^*) = (M^*, ID^*)$ where dk^* denotes the decryption key corresponding to Λ^*;
  - *no* KeyGen query was made on ID^*;
  - *no* Signcrypt query was made on M^* and ID^*.

We define the advantage of the adversary as

$$\mathtt{Adv}_{\mathcal{A}}(k) = \Pr[\mathcal{A}\ wins].$$

Definition 3. *We say that the HSC scheme is* existentially unforgeable under chosen message attacks (EUF-CMA) *if for any PPT $\mathcal{A}$, $\mathtt{Adv}_{\mathcal{A}}(k)$ is a* negligible *function of the security parameter k.*

3.3 A Novel Hybrid Signcryption Scheme

We construct a hybrid signcryption scheme based on the KP-ABE scheme proposed in [17]. We define the Lagrange coefficient $\Delta_{i,N}$ for $i \in \mathbb{Z}_q$ and a set, N, of elements in $\mathbb{Z}_q$: $\Delta_{i,N}(x) = \prod_{i \in N}^{j \neq i} \frac{x-j}{i-j}$. The data will be signcrypted under a set δ of n elements of $\mathbb{Z}_q$. The proposed hybrid signcryption scheme works as follows:

- Setup: It takes the security parameter k as input, outputs the master public key $mpk = (g, T_1 = g^{t_1}, \cdots, T_{n+1} = g^{t_{n+1}}, g^{\alpha}, g^{\beta}, e(g,h)^{\alpha}, h \in_R \mathbb{G})$ and the master secret key $msk = (t_1, \cdots, t_{n+1}, \alpha, \beta \in_R \mathbb{Z}_q)$. It also generates hash functions $\mathtt{H}_1 : \{0,1\}^* \rightarrow \mathbb{G}, \mathtt{H}_2 : \mathbb{G}_1 \rightarrow \mathbb{G}$ and chooses a pseudo-random generator G. We let N be the set $\{1, 2, \cdots n+1\}$ and denote the bilinear pairing $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. We define a function T as $T(X) = h^{X^n} \cdot \prod_{i=1}^{n+1} \cdot T_i^{\Delta_{i,N}(X)}$.

– KeyGen: It takes the identity $ID \in \{0,1\}^*$, the access tree Λ as input, outputs the signing key $sk = \mathtt{H}_1(ID)^\beta$ and the decryption key dk.
  1. It chooses a polynomial q_x for each node x (including the leaf nodes) in the tree Λ. These polynomials are chosen in the following way in a top-down manner, starting from the root node R. For each node x in the tree, set the degree d_x of the polynomial q_x to be one less than the threshold value k_x of that node (i.e., $d_x = k_x - 1$). Starting with the root note R, the algorithm will set $q_R(0) = \alpha$. Then it chooses d_R other points of the polynomial q_R randomly to define it completely. For other nodes x, it sets $q_x(0) = q_{parent(x)}(index(x))$ and chooses d_x other points randomly to completely define q_x. We define L as the set of *leaf* nodes in Λ, and proceed as follows:
  2. $\forall l \in L : D_l = h^{q_x(0)} \cdot T(i)^{r_l}, R_l = g^{r_l}$, where $i = att(l)$ and $r_l \in_R \mathbb{Z}_q$ is corresponding to leaf node l in Λ. Set $dk = \{D_l, R_l\}_{l \in L}$.

– Signcrypt: It takes a message $m \in \mathbb{G}_1$ and a set of attributes δ as input, then
  1. Computes $\widehat{C} = (m \| ID) \oplus \mathsf{G}(e(g,h)^{\alpha \cdot s}), C = g^s$, $\{C_i = T(i)^s\}_{i \in \delta}$, where $s \in_R \mathbb{Z}_q$;
  2. Computes $S = sk \cdot \mathtt{H}_2(m)^s$;
  3. Outputs the signcryption: $CT = \{\delta, \widehat{C}, C, \{C_i\}_{i \in \delta}, S\}$.

– Unsigncrypt:
  1. We define a recursive algorithm $\mathsf{DecryptNode}(CT, dk, x)$, such that dk is associated with an access tree Λ and a node x from Λ.
     - If x is a leaf node, we let $i = att(x)$.
       * If $i \in \delta$, compute

$$\begin{aligned}\mathsf{DecryptNode}(CT, dk, x) &= \frac{e(D_x, C)}{e(R_x, C_i)} \\ &= \frac{e(h^{q_x(0)} \cdot T(i)^{r_x}, g^s)}{e(g^{r_x}, T(i)^s)} \\ &= \frac{e(h^{q_x(0)}, g^s) \cdot e(T(i)^{r_x}, g^s)}{e(g^{r_x}, T(i)^s)} \\ &= e(g,h)^{s \cdot q_x(0)};\end{aligned}$$

       * If $i \notin \delta$, `abort`.
     - If x is a non-leaf node, for all nodes z, which are children of x, call $\mathsf{DecryptNode}(CT, dk, z)$ and store the output as C_z. Let S_x be an arbitrary k_x-sized set of child nodes z such that $C_z \neq \perp$. If no such set exists, the node is not satisfied and the algorithm aborts. Otherwise, compute:

$$
\begin{aligned}
C_x &= \prod_{z \in S_x} (e(g,h)^{s \cdot q_z(0)})^{\Delta_{i,S'_x}(0)} \\
&= \prod_{z \in S_x} (e(g,h)^{s \cdot q_{parent(z)}(index(z))})^{\Delta_{i,S'_x}(0)} \\
&= \prod_{z \in S_x} e(g,h)^{s \cdot q_x(i) \cdot \Delta_{i,S'_x}(0)} \\
&= e(g,h)^{s \cdot q_x(0)}.
\end{aligned}
$$

Note that $i = index(z)$, $S'_x = \{index(z) : z \in S_x\}$, and the computation $\Delta_{i,S'_x}(0)$ is computed via the polynomial interpolation according to access tree Λ.

2. If the attribute set associated with the ciphertext satisfies the tree Λ, we get $e(g,h)^{s \cdot q_R(0)} = e(g,h)^{\alpha \cdot s}$, and next compute $(m \| ID) = \widehat{C} \oplus \mathsf{G}(e(g,h)^{\alpha \cdot s})$;
3. If $e(S,g) = e(\mathtt{H}_1(ID), g^\beta) \cdot e(\mathtt{H}_2(m), C)$, it returns m; Otherwise, reject.

Lemma 1. *The proposed HSC scheme achieves EUF-CMA security under the CDH assumption.*

Proof. Let $\mathcal{S}_{CDH}$ denotes a Computational Diffie-Hellman problem solver, who is given g, g^a, g^b and aims to find g^{ab}. Let $\mathcal{A}$ denotes a forger against the proposed HSC scheme. $\mathcal{S}_{CDH}$ plays the EUF-CMA security game with $\mathcal{A}$ as follows.

- Setup Stage: Let $\mathcal{K}$ denotes the maximum number of users that will occur in the game. $\mathcal{S}_{CDH}$ randomly selects two indices i and j and guesses that the **Forge** event will happen with regard to user i and the j-th query (denote it by m^*) to the random oracle $\mathtt{H}_2$. $\mathcal{S}_{CDH}$ further sets $g^\beta = g^a$, and generates master secret/public keys $(msk = (\alpha, t_1, \cdots, t_{\mathcal{K}+1}), mpk = (T_1 = g^{t_1}, \cdots, T_{\mathcal{K}+1} = g^{t_{\mathcal{K}+1}}, g^\alpha, e(g,h)^\alpha, h = g^\theta)$ as in the real scheme. $\mathcal{S}_{CDH}$ finally sends mpk to $\mathcal{A}$. Note that $\theta \in_R \mathbb{Z}_q$ is chosen by $\mathcal{S}_{CDH}$.
- $\mathcal{S}_{CDH}$ answers $\mathcal{F}$'s queries as follows.
  - If $\mathcal{A}$ issues ID_i to random oracle $\mathtt{H}_1$, then $\mathcal{S}_{CDH}$ chooses $b_i \in_R \mathbb{Z}_q$ and returns $g^b \cdot g^{b_i}$ as the public key of user i; Otherwise, $\mathcal{S}_{CDH}$ chooses $b_j \in_R \mathbb{Z}_q$ returns the value g^{b_j} to $\mathcal{F}$.
  - If $\mathcal{A}$ queries the random oracle $\mathtt{H}_2$ with regard to the message m^*, then $\mathcal{S}_{CDH}$ chooses $c_i \in_R \mathbb{Z}_q$ and returns g^{c_i} as the response to $\mathtt{H}_2(m^*)$. If $\mathcal{A}$ queries the random oracle $\mathtt{H}_2$ with regard to other messages (e.g., m_i), then $\mathcal{S}_{CDH}$ chooses $c_i \in_R \mathbb{Z}_q$ and returns $g^{c_i - a}$ as the response.
  - If $\mathcal{A}$ issues a KeyGen query for the user i, abort. If $\mathcal{A}$ issues a KeyGen query of a user ID_j (whereby $j \neq i$), $\mathcal{S}_{CDH}$ returns the value $g^{a \cdot b_j}$ as the signing key to $\mathcal{A}$, and further simulates the decryption key exactly same as in the algorithm KeyGen. $\mathcal{A}$ is given both the signing key and the decryption key of the user j.

- $\mathcal{S}_{CDH}$ simulates the Signcrypt oracle for the user i as follows. Firstly, $\mathcal{S}_{CDH}$ chooses $k_i \in_R \mathbb{Z}_q$, generates $\widehat{C} = (m_i \| ID_i) \oplus \mathsf{G}(e(g,h)^{\alpha \cdot (b+k_i)}), C = g^{b+k_i}, S_i = g^{b \cdot c_i} \cdot g^{a \cdot (b_i - k_i)} \cdot g^{c_i \cdot k_i}$, where the randomness s is implicitly sets as $b + k_i$. Secondly, $\mathcal{S}_{CDH}$ generates $\{C_i\}_{i \in \delta} = \{T(i)^{b+k_i}\}_{i \in \delta}$ using the knowledge of t_i and θ. Finally, $\mathcal{S}_{CDH}$ returns the signcryption $CT = \{\delta, \widehat{C}, C, \{C_i\}_{i \in \delta}\}, S_i\}$ to $\mathcal{A}$. One can verify that the signcryption is valid since $g^{c_i - a} = \mathtt{H}_2(m_i)$ and $e(S_i, g) = e(g^b \cdot g^{b_i}, g^a) \cdot e(g^{c_i - a}, g^{b+k_i})$.
  Note that if $\mathcal{A}$ issues the Signcrypt query for other users, e.g., user l ($l \neq i$), $\mathcal{S}_{CDH}$ can simulate it perfectly since the simulator knows the signing key.
- If $\mathcal{A}$ issues an Unsigncrypt query, $\mathcal{S}_{CDH}$ answers the query as usual since $\mathcal{S}_{CDH}$ has the knowledge of α.

– If $\mathcal{A}$ successfully forges a signcryption CT^* including a valid forgery $C^* = g^{s^*}, S^* = g^{(b+b_i) \cdot a} \cdot \mathtt{H}_2(m^*)^{s^*}$ (notice that the randomness s^* is chosen by $\mathcal{A}$) satisfying the validity check, $\mathcal{S}_{CDH}$ can compute $g^{a \cdot b} = \dfrac{S^*}{C^{* \cdot c_i} \cdot g^{a \cdot b_i}}$ (c_i is known to $\mathcal{S}_{CDH}$ who programmed the random oracle $g^{c_i} = \mathtt{H}_2(m^*)$) as the solution of the Computational Diffie-Hellman problem.

Probability analysis: Let q_{h_i} denotes the number of queries that $\mathcal{F}$ asks to the random oracles $\mathtt{H}_i$, $i = 1, 2$. If $\mathcal{S}_{CDH}$ guesses the challenge user i and challenge message correctly, then the simulation is perfect. Therefore we have

$$\Pr[\mathbf{Forge}] \leq \mathcal{K} \cdot q_{h_2} \cdot \mathtt{Adv}_{\mathcal{S}}^{CDH}(k).$$

3.4 Our OAKE Protocol

Now we present our proposed one-round authenticated key exchange protocol in the multiple party setting. It works as follows:

– Setup: TA takes the security parameter k and the number of users $\mathcal{K}$ as input, outputs the master public key $mpk = (T_1 = g^{t_1}, \cdots, T_{n+1} = g^{t_{n+1}}, g^{\alpha}, g^{\beta}, e(g,h)^{\alpha}, \{\mathbb{G}, \mathbb{G}_1, \cdots, \mathbb{G}_{\mathcal{K}}\}, \{g, g_1, \cdots, g_{\mathcal{K}}\}, h \in_R \mathbb{G})$ and the master secret key $msk = (t_1, \cdots, t_{n+1}, \alpha, \beta \in_R \mathbb{Z}_q)$. In addition, let $\hat{\mathsf{e}} : \mathbb{G}_i \times \mathbb{G}_j \to \mathbb{G}_{i+j}$ denotes the $\mathcal{K}$-linear maps and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ denotes the bilinear maps. TA also generates three hash functions $\mathtt{H}_1 : \{0,1\}^* \to \mathbb{G}, \mathtt{H}_2 : \{0,1\}^* \to \mathbb{Z}_q, \mathtt{H}_3 : \{0,1\}^* \to \mathbb{G}$. We let N be the set $\{1, 2, \cdots n+1\}$ and define a function T as $T(X) = h^{X^n} \cdot \prod_{i=1}^{n+1} \cdot T_i^{\Delta_{i,N}(X)}$.
– KeyGen: Run the KeyGen algorithm described in the HSC scheme.
– KeyExchange: User i performs the following steps.
  1. Choose the ephemeral secret key $r_i \in_R \mathbb{Z}_q$, computes $x_i = \mathtt{H}_2(r_i \| dk_i \| sk_i)$ and $m_i = g_1^{x_i}$;
  2. Run the Signcrypt algorithm described in the HSC scheme, but the algorithm sets the randomness as x_i;
  3. Compute $S_i = sk \cdot \mathtt{H}_3(m_i || ts_i)^{x_i}$, where $ts_i \in_R \mathbb{Z}_q$ is the current timestamp generated by user i;
  4. **Broadcast** the signcryption: $CT = \{ts_i, \delta_i, \widehat{C}, C, \{C_i\}_{i \in \delta}, S_i\}$.

– SharedKey: After receiving the ciphertext $CT_j = \{ts_j, \delta_j, \widehat{C}, C, \{C_j\}_{j\in\delta}, S_j\}$ from user j, user i does the following operations.
  1. Check the time-stamp: If $|ts_i - ts_j| > \varrho$ (ts_i is the current time-stamp generated by user i and ϱ denotes the time window), then reject;
  2. Run the Unsigncrypt algorithm described in the HSC scheme, then get the message $m_j = g_1^{x_j}$ and ID_j, and verify: If $e(S_j, g) = e(\mathtt{H}_1(ID_j), g^{\beta}) \cdot e(\mathtt{H}_3(m_j||ts_j), C)$, it returns m_j; Otherwise, it rejects the session;
  3. **Compute** the session key: $SK_i = \hat{\mathsf{e}}(g_1^{x_1}, \cdots, g_1^{x_{i-1}}, g_1^{x_{i+1}}, \cdots, g_1^{x_{\mathcal{K}}})^{x_i} = g_{\mathcal{K}-1}^{\prod_{j=1}^{\mathcal{K}} x_j}$.

Design Rational. The proposed Hybrid Signcrypiton scheme has been used in the OAKE protocol for preventing the insider attacks in the multi-party setting. In addition, the Hybrid Signcryption scheme also ensures the user privacy in the proposed OAKE protocol. More details will be given in the full version of the paper.

4 Security Analysis

Theorem 1. *The proposed OAKE protocol achieves session key security (Definition 1) if the* $\mathcal{K}$-MDDH *assumption hold in the underlying group* $\mathbb{G}_{\mathcal{K}-1}$*, the proposed signcryption scheme HSC is EUF-CMA secure.*

Proof. We define a sequence of games G_i, $i = 0, \cdots, 3$ and let $\mathtt{Adv}_i^{OAKE}$ denotes the advantage of the adversary in game G_i. Assume that $\mathcal{A}$ activates at most m sessions in each game.

– G_0 This is original game $Game_{\mathcal{A}}^{OAKE}$ for session key security.
– G_1 This game is identical to game G_0 except that $\mathcal{S}$ will output a random bit if **Forge** event happens where $\mathcal{A}$ makes a send query in the form of CT_i, such that S_i is a valid signature of user i who is not corrupted (i.e., no long term key reveal query to user i or master secret key reveal query) when the send query is made, and S_i is not previously generated by the simulator. Therefore we have:

$$\left|\mathtt{Adv}_0^{OAKE} - \mathtt{Adv}_1^{OAKE}\right| \leq \Pr[\mathbf{Forge}] \tag{1}$$

Lemma 2. *The* ***Forge*** *event happens only with a negligible probability when our proposed signcryption scheme HSC is EUF-CMA secure.*

Let $\mathcal{F}$ denotes a forger against signcryption scheme HSC with EUF-CMA security, who has access to the Signcrypt oracle, the Unsigncrypt oracle and the KeyGen oracle, and aims to forge a valid signature S^*. $\mathcal{F}$ simulates the game for $\mathcal{A}$ as follows.

- Setup Stage: $\mathcal{F}$ sets up the game for $\mathcal{A}$ by creating $\mathcal{K}$ users with the corresponding identity set $\prod_{i=1}^{\mathcal{K}}\{ID_i\}$. $\mathcal{F}$ randomly selects an index i and guesses that the **Forge** event will happen with regard to user i. $\mathcal{F}$ then sends the

master public keys and the identity set to $\mathcal{A}$. $\mathcal{F}$ obtains all the user secret keys except the secret key of ID_i via the KeyGen oracle. It is obvious that $\mathcal{F}$ can answer all the queries made by $\mathcal{A}$ except user i. Below we mainly focus on the simulation of user i only.

- $\mathcal{F}$ answers $\mathcal{A}$'s queries as follows.
  - If $\mathcal{A}$ issues a send query in the form of a signcryption CT to user i, then $\mathcal{F}$ will perform the simulation as follows: $\mathcal{F}$ firstly can get the message g_1^x and the identity ID after submitting the received signcryption CT to his Unsigncrypt oracle. If $\mathcal{A}$ makes a send query in the form of an activation request, $\mathcal{F}$ randomly chooses $r_i \in_R \mathbb{Z}_q$ and programs the $\mathtt{H}_2$ oracle to get x_i, computes the message $g_1^{x_i}$ and generates the signcryption CT_i using his Signcrypt oracle on the message $g_1^{x_i}||ID_i$ and returns CT_i to $\mathcal{A}$.
  - If $\mathcal{A}$ issues an ephemeral secret key reveal query to user i, then $\mathcal{F}$ returns $r_i \in_R \mathbb{Z}_q$ to $\mathcal{A}$.
  - If $\mathcal{A}$ issues long term secret key reveal query to user i or master secret key reveal query, then $\mathcal{F}$ aborts.
  - $\mathcal{F}$ answers the session key reveal query and test query by using the session key it has derived during the protocol simulation described above.
- If a **Forge** event with respect to user i occurs, then $\mathcal{F}$ outputs whatever $\mathcal{A}$ outputs as its own forgery; Otherwise, $\mathcal{F}$ aborts the game. Therefore we have:

$$\Pr[\mathbf{Forge}] \leq \mathcal{K} \cdot \mathtt{Adv}_{\mathcal{F}}^{HSC}(k) \tag{2}$$

– G_2: This game is identical to game 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 occurred in the g-th session (denote this event by **Guess**). Therefore we have

$$\mathtt{Adv}_1^{OAKE} = m \cdot \mathtt{Adv}_2^{OAKE} \tag{3}$$

– G_3 This game is identical to game G_2 except that in the test session, we replace the session key $SK = g_{\mathcal{K}-1}^{\prod_{j=1}^{\mathcal{K}} x_j}$ by a random value $R \in_R \mathbb{G}_{\mathcal{K}-1}$. Below we show that the difference between G_2 and G_3 is negligible under the $\mathcal{K}$-MDDH assumption is hold in the group $G_{\mathcal{K}-1}$.
Let $\mathcal{S}_{\mathcal{K}-MDDH}$ denotes a distinguisher against the $\mathcal{K}$-MDDH assumption, who is given $(g_1^{c_1}, \cdots, g_1^{c_{\mathcal{K}}})$ and aims to distinguish the value $T = g_{\mathcal{K}-1}^{\prod_{j=1}^{\mathcal{K}} c_j}$ from a random value $R \in_R \mathbb{G}_{\mathcal{K}-1}$. $\mathcal{S}_{\mathcal{K}-MDDH}$ simulates the game for $\mathcal{A}$ as follows.
  - Setup Stage: $\mathcal{S}_{\mathcal{K}-MDDH}$ sets up the game for $\mathcal{A}$ by creating $\mathcal{K}$ users. $\mathcal{S}_{\mathcal{K}-MDDH}$ then generates the master public/secret key pair (mpk, msk) and the secret keys $\{(dk_i, sk_i)\}$ for all the users, where the dk_i is corresponding to an access tree Λ_i and sk_i is corresponding to the identity ID_i of user i. $\mathcal{S}_{\mathcal{K}-MDDH}$ then sends the master public key and the identity set to $\mathcal{A}$.
  - It is easy to see that all queries to a user can be simulated perfectly using the user secret keys. In the g-th (i.e., test) session, $\mathcal{S}_{\mathcal{K}-MDDH}$ sets $m_1 =$

$g_1^{c_1}, \cdots, m_{\mathcal{K}} = g_1^{c_{\mathcal{K}}}$ for all the users which implicitly sets $H(r_i \| dk_i \| sk_i) = c_i$ where r_i denotes the ephemeral key of user i in the g-th session. Since $\mathcal{A}$ is not allowed to ask both ephemeral and long term secret keys of a user in the test session, the simulation is perfect.

- $\mathcal{S}_{\mathcal{K}-MDDH}$ answers the Test query by using its own challenge as the session key of the g-th session.
- If $\mathcal{A}$ wins the game, then $\mathcal{S}_{\mathcal{K}-MDDH}$ outputs that the challenge is $g_{\mathcal{K}-1}^{\prod_{j=1}^{\mathcal{K}} c_j}$; Otherwise $\mathcal{S}_{\mathcal{K}-MDDH}$ outputs that the challenge is a random element.

If the challenge of $\mathcal{S}_{\mathcal{K}-MDDH}$ is $g_{\mathcal{K}-1}^{\prod_{j=1}^{\mathcal{K}} c_j}$, then the simulation is consistent with G_2; Otherwise, the simulation is consistent with G_3. If the advantage of $\mathcal{A}$ is significantly different in G_2 and G_3, then $\mathcal{S}_{\mathcal{K}-MDDH}$ can break the $\mathcal{K}$-MDDH assumption. Therefore we have

$$\left|\mathtt{Adv}_2^{OAKE} - \mathtt{Adv}_3^{OAKE}\right| \leq \mathtt{Adv}_{\mathcal{S}_{\mathcal{K}-MDDH}}^{\mathcal{K}-MDDH}(k) \tag{4}$$

It is easy to see that in game G_3, $\mathcal{A}$ has no advantage, i.e.,

$$\mathtt{Adv}_3^{OAKE} = 0 \tag{5}$$

Combining the above results together, we have

$$\mathtt{Adv}_{\mathcal{A}}^{OAKE}(k) \leq \mathcal{K} \cdot \mathtt{Adv}_{\mathcal{F}}^{HSC}(k) + m \cdot \mathtt{Adv}_{\mathcal{S}_{\mathcal{K}-MDDH}}^{\mathcal{K}-MDDH}(k)$$

Theorem 2. *The proposed OAKE protocol achieves insider security (Definition 2) if the proposed signcryption scheme HSC is EUF-CMA secure.*

The proof of insider security can be obtained from the proof of Lemma 2 since if an attacker can break the insider security with a non-negligible probability, then a **Forge** event would occur also with a non-negligible probability. We omit the details of the proof here.

5 Conclusion

In this paper, we proposed a one-round attribute-based key exchange (OAKE) protocol in the multi-party setting. In order to address the insider security issue, we proposed a new primitive named hybrid signcryption which is a combination of attribute-based encryption and identity-based signature. We used this new primitive and the multilinear maps as major building block in constructing our OAKE protocol. We also defined the formal security models for session key security and insider security, and proved the security of the proposed OAKE protocol in the random oracle model.

References

1. An, J.H., Dodis, Y., Rabin, T.: On the security of joint signature and encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 83–107. Springer, Heidelberg (2002). doi:10.1007/3-540-46035-7_6

2. Baek, J., Steinfeld, R., Zheng, Y.: Formal proofs for the security of signcryption. J. Cryptology **20**(2), 203–235 (2007)
3. Bellare, M., Canetti, R., Krawczyk, H.: A modular approach to the design and analysis of authentication and key exchange protocols (extended abstract). In: Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, pp. 419–428 (1998)
4. Bellare, M., Pointcheval, D., Rogaway, P.: Authenticated key exchange secure against dictionary attacks. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 139–155. Springer, Heidelberg (2000). doi:10.1007/3-540-45539-6_11
5. Bellare, M., Rogaway, P.: Entity authentication and key distribution. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 232–249. Springer, Heidelberg (1994). doi:10.1007/3-540-48329-2_21
6. Bellare, M., Rogaway, P.: Provably secure session key distribution: the three party case. In: Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing, pp. 57–66 (1995)
7. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: 2007 IEEE Symposium on Security and Privacy (S&P 2007), pp. 321–334 (2007)
8. Bohli, J., Vasco, M.I.G., Steinwandt, R.: Secure group key establishment revisited. Int. J. Inf. Secur. **6**(4), 243–254 (2007)
9. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001). doi:10.1007/3-540-44647-8_13
10. Boneh, D., Waters, B.: Constrained pseudorandom functions and their applications. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, vol. 8270, pp. 280–300. Springer, Heidelberg (2013). doi:10.1007/978-3-642-42045-0_15
11. Boyen, X.: Multipurpose identity-based signcryption. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 383–399. Springer, Heidelberg (2003). doi:10.1007/978-3-540-45146-4_23
12. Burmester, M., Desmedt, Y.: Efficient and secure conference-key distribution. In: Security Protocols, International Workshop, Cambridge, United Kingdom, 10–12 April 1996, p. 119–129 (1996)
13. Canetti, R., Krawczyk, H.: Analysis of key-exchange protocols and their use for building secure channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001). doi:10.1007/3-540-44987-6_28
14. Cha, J.C., Cheon, J.H.: An identity-based signature from gap diffie-hellman groups. In: IACR Cryptology ePrint Archive 2002, vol. 18 (2002)
15. Gagné, M., Narayan, S., Safavi-Naini, R.: Threshold attribute-based signcryption. In: Security and Cryptography for Networks, pp. 154–171 (2010)
16. Gorantla, M.C., Boyd, C., González Nieto, J.M.: Attribute-based authenticated key exchange. In: Steinfeld, R., Hawkes, P. (eds.) ACISP 2010. LNCS, vol. 6168, pp. 300–317. Springer, Heidelberg (2010). doi:10.1007/978-3-642-14081-5_19
17. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM, CCS 2006, pp. 89–98 (2006)
18. Haber, S., Pinkas, B.: Securely combining public-key cryptosystems. In: CCS 2001, pp. 215–224 (2001)
19. Hess, F.: Efficient identity based signature schemes based on pairings. In: Nyberg, K., Heys, H. (eds.) SAC 2002. LNCS, vol. 2595, pp. 310–324. Springer, Heidelberg (2003). doi:10.1007/3-540-36492-7_20
20. Katz, J., Shin, J.S.: Modeling insider attacks on group key-exchange protocols. In: ACM, CCS 2005, pp. 180–189 (2005)

21. Katz, J., Yung, M.: Scalable protocols for authenticated group key exchange. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 110–125. Springer, Heidelberg (2003). doi:10.1007/978-3-540-45146-4_7
22. Krawczyk, H.: HMQV: a high-performance secure diffie-hellman protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (2005). doi:10.1007/11535218_33
23. LaMacchia, B.A., Lauter, K.E., Mityagin, A.: Stronger security of authenticated key exchange. In: Provable Security 2007, pp. 1–16 (2007)
24. Rao, Y.S., Dutta, R.: *Expressive* bandwidth-efficient attribute based signature and signcryption in standard model. In: Susilo, W., Mu, Y. (eds.) ACISP 2014. LNCS, vol. 8544, pp. 209–225. Springer, Heidelberg (2014). doi:10.1007/978-3-319-08344-5_14
25. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005). doi:10.1007/11426639_27
26. Yoneyama, K.: Strongly secure two-pass attribute-based authenticated key exchange. In: Joye, M., Miyaji, A., Otsuka, A. (eds.) Pairing 2010. LNCS, vol. 6487, pp. 147–166. Springer, Heidelberg (2010). doi:10.1007/978-3-642-17455-1_10
27. Zheng, Y.: Digital signcryption or how to achieve cost(signature & encryption) $\ll$ cost(signature) + cost(encryption). In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 165–179. Springer, Heidelberg (1997)