1-1-2014

Attribute-based optimistic fair exchange: how to restrict brokers with policies

Yang Wang
University of Wollongong, yw990@uowmail.edu.au

Man Ho Allen Au
University of Wollongong, aau@uow.edu.au

Willy Susilo
University of Wollongong, wsusilo@uow.edu.au

Attribute-based optimistic fair exchange: how to restrict brokers with policies

Abstract

Optimistic fair exchange (OFE) is a kind of protocols for solving the fair exchange problem between two participants with the help of an arbitrator that only needs to be involved when dispute occurs. As far as we are concerned, all previous work on OFE does not take into account user's attributes such as nationality and age. We identify that in some applications, the attributes could play an important role in the exchange to take place, and OFE may not be suitable to these scenarios. We introduce a new notion named attribute-based optimistic fair exchange (ABOFE) to solve the fair exchange problem in the attribute-based setting. We formalize the notion of ABOFE and present a security model in the multi-user setting under the chosen-key attack. We also present a generic construction of ABOFE from existing cryptographic primitives and prove that our proposal is secure with respect to our denition in the standard model. An instantiation in the standard model is discussed.

Keywords

brokers, restrict, policies, exchange, attribute, fair, optimistic

Publication Details

Wang, Y., Au, M. & Susilo, W. (2014). Attribute-based optimistic fair exchange: how to restrict brokers with policies. Theoretical Computer Science, 527 83-96.

Attribute-based Optimistic Fair Exchange: How to Restrict Brokers with Policies

Yang Wang[a], Man Ho Au[a], Willy Susilo[a,1,*]

[a] *Centre for Computer and Information Security Research*
School of Computer Science and Software Engineering
University of Wollongong, Australia

Abstract

Optimistic fair exchange (OFE) is a kind of protocols for solving the fair exchange problem between two participants with the help of an arbitrator that only needs to be involved when dispute occurs. As far as we are concerned, all previous work on OFE does not take into account user's attributes such as nationality and age. We identify that in some applications, the attributes could play an important role in the exchange to take place, and OFE may not be suitable to these scenarios. We introduce a new notion named *attribute-based optimistic fair exchange* (ABOFE) to solve the fair exchange problem in the attribute-based setting. We formalize the notion of ABOFE and present a security model in the multi-user setting under the chosen-key attack. We also present a generic construction of ABOFE from existing cryptographic primitives and prove that our proposal is secure with respect to our definition in the standard model. An instantiation in the standard model is discussed.

Keywords: Optimistic fair exchange, Attribute-based encryption, Policy, Standard model

*Corresponding author.

Email addresses: yw990@uowmail.uow.edu.au (Yang Wang), aau@uow.edu.au (Man Ho Au), wsusilo@uow.edu.au (Willy Susilo)

[1]This work is supported by ARC Future Fellowship FT0991397.

1. Introduction

The brokerage business model has been used since the pre-Internet era, where the intermediary buys from the supplier (or producer) and owns the goods first, and then sells it. This model plays a very important role in the online business nowadays, as it enables fast and secure transactions without relying on a single merchant's connection. Brokers have been known to be active in business-to-business (B2B), businesss-to-consumer (B2C) and consumer-to-consumer (C2C) or peer-to-peer (P2P) markets. Although this model is known to be very useful and practical, some issues have happened since the broker may incorporate some certain strategies to increase his/her sales, and it may damage even the supplier.

Consider the following real life case study. A broker B buys games from the game developer G and sells these games to its customer. Since B is considered as a broker, the price that has been set to B is certainly lower than the retail price of the game itself. In order to maximise its sales, B is happy to make its margin to be very low, and this way B will gain popularity among the customers and attract more buyers. In particular, this is certainly possible if B purchases the games from different countries, since usually the price for each country will be different, due to the sales tax applied. This action is certainly damaging the market and G will not be able to sell that particular game with the retail price to the customers as they would have preferred to acquire it from B instead. This issue can be solved by placing required policies for the brokers. First of all, each broker must be licensed in order to sell the games. The license is limited to the country of residence. The game will be playable, if and only if, the required CD key sold by the broker matches with the country where the game will be played. To give an example, someone resides in the US will purchase a CD key from the broker. First of all, he/she needs to be sure that the broker is a licensed retailer for US. Once the CD key is issued by the broker, this CD key is only usable if it is used in the US and not in other countries. Another scenario involves limitation of the age restriction towards the game itself. For instance, consider the game "God of War" that is restricted to people over 18 years old. A US online reseller selles the activation key (or CD key) for this game. This price is lower than the price of the same game available in Canada. There are two issues need to be solved here. First, the buyer needs to be ensured that the reseller is a genuine reseller, and the reseller needs to be sure that the buyer is at least 18 years old. Subsequently, the activation key sold can only be

used in the US and not elsewhere. Hence, the buyer from Canada will not be able to make use of this particular activation key.

Our Approach. In this paper, we intend to present a cryptographic primitive that aims to solve the above scenario. We make use of the notion of optimistic fair exchange (OFE) and enhance this notion to enable policies for both parties, namely the seller and the buyer (or the signer and the verifier, resp.). A trivial solution for enabling the above scenario would be that prior to involving an OFE protocol, each participant will simply present an evidence, for instance a copy of the identity card or license, to the other participant, hence confirming that they satisfy the requirements. Nevertheless, this solution compromises the privacy of the users, and therefore it is not ideal. Furthermore, providing such kind of evidence over the network securely is rather challenging as well.

1.1. Optimistic Fair Exchange

Optimistic fair exchange (OFE), first introduced by Asokan, Schunter and Waidner [1], is a kind of protocols aiming to guarantee fairness for two parties Alice and Bob exchanging digital items. An trusted third party named "arbitrator" is needed in OFE, but involves only when there is a dispute between exchanging parties. Since a large number of digital items such as electronic checks and electronic airline tickets are implemented as digital signatures, the optimistic fair exchange of digital signatures constitutes an important part of any business transaction.

In a typical execution of optimistic fair exchange of digital signatures, Alice first sends a partial signature to Bob. Bob verifies the validity of Alice's partial signature, and sends its full signature to Alice, after which Alice sends her full signature back to Bob and completes the exchange. In the case there is a network failure or Alice attempts to cheat by refusing to send her own full signature, the arbitrator will convert Alice's partial signature into a full one and send it back to Bob. It is implicitly assumed that Bob should offer his own full signature when asking the arbitrator for help, and the arbitrator will send Bob's full signature to Alice later. Thus at the end of this exchange, either both Alice and Bob gain the other's full signature, or neither does. Thus the exchange is fair.

As a useful tool in applications such as contract signing and electronic commerce, OFE has been extensively researched [2, 3, 4, 5, 6, 7, 8] since its introduction. Several primitives are useful for the construction of OFE,

including verifiably encrypted signatures [9, 10, 11, 12, 13, 14, 15], and sequentially two-party multisignatures [16]. It was further showed that OFE can be constructed from OR signature [17], and from conventional signatures and ring signatures [18].

In an orthogonal dimension, formal OFE security models [17, 18] are proposed. While most optimistic fair exchange protocols are studied in the *certified-key* model (also known as the *registered-key* model [19]) in which the adversary is only allowed to make queries with respect to the registered public keys, the security of optimistic fair exchange in the multi-user setting and *chosen-key model* [18], where the adversary can choose its public key arbitrarily probably without knowing the corresponding private keys, is more desirable.

1.2. Our Contribution

In this paper, motivated by idea of attribute-based encryption (ABE) [20] and ciphertext policy attribute-based encryption (CPABE) [21], we introduce the notion of *attribute-based optimistic fair exchange* (ABOFE), which can be viewed as an extension of OFE, as a practical cryptographic solution to the aforementioned scenario.

In ABOFE, each user satisfying a set of attributes is assigned a credential by the credential center. This allows the signer Alice to generate a credential-protected package in such a way that only verifiers that possess appropriate credentials can convert it into a full signature, which naturally guarantees that the verifier should satisfy some particular set of attributes.

We propose the syntax and also define a security model for ABOFE in the multi-user setting under chosen-key attack. Our model captures the existing security requirements for OFE, namely, security against signers, security against verifiers and security against the arbitrator. As suggested by the respective names, they intend to cover the scenarios when the named party is dishonest.

Finally, we propose a generic construction of ABOFE from the two well established cryptographic primitives, OFE and CP-ABE, and provide the security proof of our proposal in the proposed model. Our generic construction works in the standard model and does not involve any extra assumptions. The efficiency of an instantiation about the generation construction is also discussed.

1.3. Paper Organization

In the next section, we review the notions and security models of OFE and CP-ABE, respectively. In Section 3, the syntax of ABOFE and its security definitions are presented. Then we present our construction in Section 4. An instantiation is descussed in Section 5. Finally, we conclude our paper in Section 6.

2. Preliminaries

Throughout the paper, the following notations are used. For a finite set $\mathcal{S}$, $s \leftarrow \mathcal{S}$ denotes that an element is randomly chosen from $\mathcal{S}$. By $y \leftarrow A^O(x)$, we mean the algorithm A, on input x and having access to oracle O, outputs y. By $x := y$, we mean variable x is assigned with the value of y.

Definition 1 (Access structure [22]). *Let $\{P_1, \cdots, P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,\cdots,P_n\}}$ is monotone if for any B, C: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \subseteq \mathbb{A}$. An access structure (respectively, monotonic access structure) is a collection (respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, \cdots, P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1,\cdots,P_n\}} \setminus \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called unauthorized sets.*

In this paper, the attributes will take the role of the parties and by an access structure, we mean a monotone access structure. When we say a set S satisfies an access structure $\mathbb{A}$, we mean $S \in \mathbb{A}$.

2.1. Ciphertext-Policy ABE

We review the notion and security model of ciphertext-policy attribute-based encryption introduced in [23]. A ciphertext-policy attribute-based encryption scheme $\mathcal{E}$ comprises four efficient algorithms: $\mathcal{E} := (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{KeyGen}, \mathsf{Decrypt})$.

$\mathsf{Setup}(1^k, U)$. The setup algorithm takes as input security parameter 1^k and attribute universe description U, and outputs the public parameters PM and a master key MK.

$\mathsf{Encrypt}(\mathrm{PM}, M, \mathbb{A})$. The encryption algorithm takes as input the public parameters PM, a message M, and an access structure $\mathbb{A}$ over the universe of attributes, and outputs a ciphertext CT such that only a user that possesses

a set of attributes that satisfies the access structure $\mathbb{A}$ will be able to decrypt the ciphertext. It is assumed that the ciphertext implicitly contains $\mathbb{A}$.

$\mathsf{KeyGen}(\mathrm{MK}, S)$. The key generation algorithm takes as input the master key MK and a set of attributes S, and outputs a private key SK.

$\mathsf{Decrypt}(\mathrm{PM}, \mathrm{CT}, \mathrm{SK})$. The decryption algorithm takes as input the public parameters PM, a ciphertext CT, which contains an access policy $\mathbb{A}$, and a private key SK, which is a private key for a set of attributes S. If the set of attributes S satisfies the access structure $\mathbb{A}$ then the algorithm will decrypt the ciphertext and returns a message M.

Correctness for a CP-ABE scheme $\mathcal{E}$ states that $\mathsf{Decrypt}(\mathrm{PM}, \mathsf{Encrypt}(\mathrm{PM}, M, \mathbb{A}), \mathsf{KeyGen}(\mathrm{MK}, S)) = M$, for any set of attributes S that satisfies the access structure $\mathbb{A}$.

2.1.1. Security Model.

The full security for CP-ABE [23] is described by a security game between a challenger and an adversary. The game proceeds as follows.

Setup The challenger runs Setup algorithm and sends the public parameters PM to the adversary.

Phase 1 The adversary adaptively queries the challenger for private keys corresponding to sets of attributes $S_1, \cdots, S_{q_1}$. For the query with respect to S_k , the challenger responds with a secret key outputted by $\mathsf{KeyGen}(\mathrm{MK}, S_k)$.

Challenge The adversary submits two equal length messages M_0 and M_1, and an access structure $\mathbb{A}$ such that none of the previous sets $S_1, \cdots, S_{q_1}$ satisfies the access structure. The challenger flips a random coin $b \in \{0, 1\}$, and encrypts M_b under $\mathbb{A}$, producing CT. It sends CT to the attacker.

Phase 2 The adversary adaptively queries the challenger for private keys corresponding to sets of attributes $S_{q_1+1}, \cdots, S_q$, with the restriction that none of these satisfies $\mathbb{A}$. For the query with respect to S_k, the challenger responds with a secret key outputted by $\mathsf{KeyGen}(\mathrm{MK}, S_k)$.

Guess The adversary outputs a guess $b' \in \{0, 1\}$.

The advantage of an adversary in this game is defined as $\Pr[b' = b] - \frac{1}{2}$.

Definition 2. *A ciphertext-policy attribute-based encryption scheme is fully secure if all polynomial time adversaries have at most a negligible advantage in the above security game.*

2.2. Optimistic Fair Exchange

We review the notion and security model of the optimistic fair exchange protocol introduced in [24].

Definition 3. *An optimistic fair exchange scheme involves the users (signers and verifiers) and the arbitrator, and is formalized by the following (probabilistic) polynomial-time algorithms:*

- $\mathsf{Setup}^{\mathsf{TTP}}$*: On input a security parameter* 1^k*, the algorithm generates a secret key* ASK*, and a public key* APK *of the arbitrator.*
- $\mathsf{Setup}^{\mathsf{User}}$*: On input* 1^k *and (optionally)* APK*, it outputs a secret/public key pair* (SK, PK)*. For a user* U_i*, we use* (SK_i, PK_i) *to denote the user's key pair.*
- Sig *and* Ver*:* $\mathsf{Sig}(M, SK_i, APK)$*, outputs a (full) signature* σ *on message* M *of user* U_i*, while* $\mathsf{Ver}(M, \sigma, PK_i, APK)$ *outputs* $\top$ *or* $\perp$*, indicating* σ *is* U_i*'s valid full signature on* M *or not, respectively.*
- PSig *and* PVer*: These are partial signing and partial verification algorithms respectively.* $\mathsf{PSig}(M, SK_i, APK)$ *outputs a partial signature* σ_P*, while* $\mathsf{PVer}(M, \sigma_P, PK_i, APK)$ *outputs* $\top$ *or* $\perp$*.*
- Res*: This is the resolution algorithm.* $\mathsf{Res}(M, \sigma_P, ASK, PK_i)$ *outputs a full signature* σ*, or* $\perp$ *indicating the failure of resolving a partial signature.*

Correctness property states that

- $\mathsf{Ver}(M, \mathsf{Sig}(M, SK_i, APK), PK_i, APK) = \top$,
- $\mathsf{PVer}(M, \mathsf{PSig}(M, SK_i, APK), PK_i, APK) = \top$,
- $\mathsf{Ver}(M, \mathsf{Res}(M, \mathsf{PSig}(M, SK_i, APK), ASK, PK_i), PK_i, APK) = \top$.

Resolution ambiguity property states that

- *any "resolved signature"* $\mathsf{Res}(M, \mathsf{PSig}(M, SK_i, APK), ASK, PK_i)$ *is computationally indistinguishable from the "actual signature"* $\mathsf{Sig}(M, SK_i, APK)$.

The security of an optimistic fair exchange scheme consists of three aspects: security against signers, security against verifiers, and security against the arbitrator. The security models of them in the multi-user setting and chosen-key model are reviewed below.

SECURITY AGAINST SIGNERS. We require that any PPT adversary $\mathcal{A}$ succeeds with at most negligible probability in the following experiment.

$$\begin{aligned}
(ASK, APK) &\leftarrow \mathsf{Setup}^{\mathsf{TTP}}(1^k) \\
(M, \sigma_P, PK_A) &\leftarrow \mathcal{A}^{O_{\mathsf{Res}}}(APK) \\
\sigma &\leftarrow \mathsf{Res}(M, \sigma_P, ASK, PK_A) \\
\text{success of } \mathcal{A} &:= [\mathsf{PVer}(M, \sigma_P, PK_A, APK) = \top \\
&\quad \wedge \mathsf{Ver}(M, \sigma, PK_A, APK) = \bot]
\end{aligned}$$

where the resolution oracle O_{Res} takes as input a valid partial signature σ_P of user U_i on message M (i.e. (M, σ_P, PK_i) such that $\mathsf{PVer}(M, \sigma_P, PK_i, APK) = \top$) and outputs a full signature σ on M under PK_i. In other words, no signer should be able to produce a partial signature that looks good to a verifier but cannot be resolved to a full signature by the honest arbitrator.

SECURITY AGAINST VERIFIERS. We require that any PPT adversary $\mathcal{A}$ succeeds with at most negligible probability in the following experiment.

$$\begin{aligned}
(ASK, APK) &\leftarrow \mathsf{Setup}^{\mathsf{TTP}}(1^k) \\
(SK_A, PK_A) &\leftarrow \mathsf{Setup}^{\mathsf{User}}(APK) \\
(M, \sigma) &\leftarrow \mathcal{A}^{O_{\mathsf{PSig}}, O_{\mathsf{Res}}}(APK, PK_A) \\
\text{success of } \mathcal{A} &:= [\mathsf{Ver}(M, \sigma, PK_A, APK) = \top \\
&\quad \wedge (M, \cdot, PK_A) \notin Query(\mathcal{A}, O_{\mathsf{Res}})]
\end{aligned}$$

where oracle O_{Res} is described in the previous experiment, the partial signing oracle O_{PSig} takes a message M as input and outputs a partial signature σ_P on M under PK_A, and $Query(\mathcal{A}, O_{\mathsf{Res}})$ is the set of queries made by $\mathcal{A}$ to oracle O_{Res}. In other words, no verifier should be able to complete any partial signature σ_P into a full signature, without explicitly asking the arbitrator to do so. Note that there is no need to provide $\mathcal{A}$ with the signing oracle O_{Sig},

as the resolution ambiguity property guarantees that it can be simulated by O_{PSig} and O_{Res}.

SECURITY AGAINST THE ARBITRATOR. We require that any PPT two-stage adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ succeeds with at most negligible probability in the following experiment.

$$\begin{aligned}
(APK, ASK^*) &\leftarrow \mathcal{A}_1(1^k) \\
(SK_A, PK_A) &\leftarrow \mathsf{Setup}^{\mathsf{User}}(APK) \\
(M, \sigma) &\leftarrow \mathcal{A}_2^{O_{\mathsf{PSig}}}(ASK^*, APK, PK_A) \\
\text{success of } \mathcal{A} &:= [\mathsf{Ver}(M, \sigma, PK_A, APK) = \top \\
&\quad \wedge M \notin Query(\mathcal{A}, O_{\mathsf{PSig}})]
\end{aligned}$$

where ASK^* is $\mathcal{A}_1$'s state information, which might not be the corresponding private key of APK, oracle O_{PSig} is described in the previous experiment, and $Query(\mathcal{A}, O_{\mathsf{PSig}})$ is the set of queries made by $\mathcal{A}$ to oracle O_{PSig}. In other words, the arbitrator should not be able to produce a full signature without explicitly asking the signer to generate a partial one.

3. Attribute-based Optimistic Fair Exchange

We adapt the definitions and security models of OFE from various literatures for our attribute-based OFE. In ABOFE, besides the traditional secret/public key pair, each user also possesses a credential, generated by the credential center, corresponding to a set of attributes the user satisfies. We require that a user that does not possess a set of attributes should not be able to gain other users' full signatures. Thus a full signature will not be sent by the signer in ABOFE, which is different from the case in OFE.

With this in mind, we introduce the the algorithms of which ABOFE scheme consists.

Definition 4. *An attribute-based optimistic fair exchange scheme involves the users, the credential center and the arbitrator, and consists of the following (probabilistic) polynomial-time algorithms:*

- **PMGen***: On input 1^k and attribute universe description U where k is a security parameter, this algorithm outputs a system parameter* PM *and a credential secret key* CK*.*

- $\mathbf{Setup}^{\mathsf{TTP}}$: *On input* 1^k*, the algorithm generates a secret key* ASK*, and a public key* APK *of the arbitrator.*

- $\mathbf{Setup}^{\mathsf{User}}$: *On input* 1^k *and (optionally)* APK*, it outputs a secret/public key pair* $(\mathsf{SK}, \mathsf{PK})$*. For a user* U_i*, we use* $(\mathsf{SK}_i, \mathsf{PK}_i)$ *to denote the user's key pair.*

- $\mathbf{Setup}^{\mathsf{Cred}}$: *On input the credential secret key* CK *and a set of attributes* S*, the attribute key generation algorithm outputs a credential* CD_S *corresponding to the set of attributes* S*.*

- $\mathbf{Tran}_{\mathsf{P}}$: *The transaction promise generation algorithm* $\mathbf{Tran}_{\mathsf{P}}(m_1, m_2, \mathsf{SK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2)$ *takes as input two messages* m_1 *and* m_2*, respectively, user* U_i*'s secret key* SK_i*, the arbitrator's public key* APK*, the system parameter* PM *and two access structure* $\mathbb{A}_1$ *and* $\mathbb{A}_2$ *over the universe of attributes, respectively. This algorithm outputs a transaction promise* ω*.*

- $\mathbf{TPVer}$: *The transaction promise verification algorithm* $\mathbf{TPVer}(m_1, m_2, \omega, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2)$ *takes as input two message* m_1 *and* m_2*, respectively, a transaction promise* ω*, user* U_i*'s public key* PK_i*, the arbitrator's public key* APK*, the system parameter* PM*, and two access structure* $\mathbb{A}_1$ *and* $\mathbb{A}_2$*, respectively. The algorithm outputs* $\top$ *indicating valid or* $\perp$ *indicating invalid.*

- $\mathbf{Tran}_{\mathsf{S}}$: *The transaction for signer algorithm* $\mathbf{Tran}_{\mathsf{S}}(m, \mathsf{SK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A})$ *takes as input a message* m*, user* U_i*'s secret key* SK_i*, the arbitrator's public key* APK*, the system parameter* PM*, and an access structure* $\mathbb{A}$*. The algorithm outputs a credential-protected package* π*.*

- $\mathbf{Tran}_{\mathsf{V}}$: *The transaction for verifier algorithm* $\mathbf{Tran}_{\mathsf{V}}(m, \pi, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, \mathsf{CD}_S)$ *takes a message* m*, a credential-protected package* π*, user* U_i*'s public key* PK_i*, the arbitrator's public key* APK*, the system parameter* PM*, and a credential* CD_S*. This algorithm outputs a full signature* σ *or* $\perp$ *indicating failure.*

- **Ver:** *The full signature verification algorithm* $\mathbf{Ver}(m, \sigma, \mathsf{PK}_i, \mathsf{APK})$ *outputs* $\top$ *indicating valid or* $\perp$ *indicating invalid.*

- $\mathbf{Res}_{\mathsf{User}}$: *The resolution for user algorithm* $\mathbf{Res}_{\mathsf{User}}(m_1, m_2, \omega, \mathsf{SK}_j, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2)$ *takes as input two message* m_1 *and* m_2*, respectively,*

a transaction promise ω, and user U_j's secret key SK_j, the system parameter PM, and two access structure $\mathbb{A}_1$ and $\mathbb{A}_2$, respectively. This algorithm outputs a resolution request request.

- $\mathbf{Res}_{\mathsf{TTP}}$*: The resolution for arbitrator algorithm* $\mathbf{Res}_{\mathsf{TTP}}(m, \mathsf{PK}_i, \mathbb{A},$ $\mathsf{request}, \mathsf{ASK})$ *takes as input input a message m, user U_i's public key PK_i, an access structure $\mathbb{A}$, a resolution request* request *and the arbitrator's secret key* ASK*. It outputs two credential-protected packages π_1 and π_2, respectively.*

Correctness states that, for $\mathbf{PMGen}(1^k) \rightarrow (\mathsf{PM}, \mathsf{CK})$, $\mathbf{Setup}^{\mathsf{TTP}}(1^k) \rightarrow (\mathsf{ASK}, \mathsf{APK})$, $\mathbf{Setup}^{\mathsf{User}}(1^k) \rightarrow (\mathsf{SK}_i, \mathsf{PK}_i)$, $\mathbf{Setup}^{\mathsf{User}}(1^k) \rightarrow (\mathsf{SK}_j, \mathsf{PK}_j)$, $\mathbf{Tran}_{\mathsf{P}}$ $(m_1, m_2, \mathsf{SK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2) \rightarrow \omega$, $\mathbf{Tran}_{\mathsf{S}}$ $(m_1, \mathsf{SK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_2) \rightarrow \pi_1$, $\mathbf{Tran}_{\mathsf{S}}$ $(m_2, \mathsf{SK}_j, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1) \rightarrow \pi_2$, $\mathbf{Res}_{\mathsf{User}}$ $(m_1, m_2, \omega, \mathsf{SK}_j, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2) \rightarrow \mathsf{request}$, $\mathbf{Res}_{\mathsf{TTP}}(m_1, \mathsf{PK}_i, \mathbb{A}_2, \mathsf{request}, \mathsf{ASK}) \rightarrow (\tilde{\pi}_1, \tilde{\pi}_2)$, $\mathbf{Setup}^{\mathsf{Cred}}$ $(\mathsf{CK}, S) \rightarrow \mathsf{CD}_S$ where the sets of attributes S satisfies the access structures $\mathbb{A}_1$, and $\mathbf{Setup}^{\mathsf{Cred}}(\mathsf{CK}, S') \rightarrow \mathsf{CD}_{S'}$ where the sets of attributes S' satisfies the access structures $\mathbb{A}_2$, the following conditions hold:

- $\mathbf{TPVer}(m_1, m_2, \omega, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2) = \top$.
- $\mathbf{Ver}(m_1, \mathbf{Tran}_{\mathsf{V}}(m_1, \pi_1, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, \mathsf{CD}_{S'}), \mathsf{PK}_i, \mathsf{APK}) = \top$.
- $\mathbf{Ver}(m_2, \mathbf{Tran}_{\mathsf{V}}(m_2, \pi_2, \mathsf{PK}_j, \mathsf{APK}, \mathsf{PM}, \mathsf{CD}_S), \mathsf{PK}_j, \mathsf{APK}) = \top$.
- $\mathbf{Ver}(m_1, \mathbf{Tran}_{\mathsf{V}}(m_1, \tilde{\pi}_1, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, \mathsf{CD}_{S'}), \mathsf{PK}_i, \mathsf{APK}) = \top$.
- $\mathbf{Ver}(m_2, \mathbf{Tran}_{\mathsf{V}}(m_2, \tilde{\pi}_2, \mathsf{PK}_j, \mathsf{APK}, \mathsf{PM}, \mathsf{CD}_S), \mathsf{PK}_j, \mathsf{APK}) = \top$.

Resolution ambiguity states that, any "resolved credential-protected packages" $\tilde{\pi}_1$ and $\tilde{\pi}_2$ outputted by $\mathbf{Res}_{\mathsf{Abr}}$ are computationally indistinguishable from the "actual credential-protected packages" π_1 and π_2 generated by $\mathbf{Tran}_S$, respectively.

3.1. A Typical Usage of ABOFE

For simplicity, we will describe a typical usage of ABOFE between two users Alice and Bob. Before the exchange phase, the credential center runs **PMGen** to set up the public parameters and keep the credential secret key as private. Alice and Bob acquire their respective credentials generated by $\mathbf{Setup}^{\mathsf{Cred}}$ from the credential center. Besides, the arbitrator sets its key

pair by invoking $\mathbf{Setup}^{\mathsf{TTP}}$. Alice and Bob generate their own key pairs by invoking $\mathbf{Setup}^{\mathsf{User}}$, respectively.

Suppose Alice intends to exchange her own full signature σ_1 on message m_1 for Bob's full signature σ_2 on message m_2. Besides, Alice should satisfy an access structure $\mathbb{A}_1$, and Bob should satisfy an access structure $\mathbb{A}_2$. The exchange phase consists of three steps.

1. Alice invokes $\mathbf{Tran}_{\mathsf{P}}$ to generate a transaction promise ω on message m_1 and m_2 with respect to access structures $\mathbb{A}_1$ and $\mathbb{A}_2$. Alice sends ω to Bob.
2. Bob invokes **TPVer** to ensure the validity of ω. Bob then invokes $\mathbf{Tran}_{\mathsf{S}}$ to generate a credential-protected package π_B on message m_2 with respect to access structure $\mathbb{A}_1$. Bob sends π_B to Alice.
3. If satisfying the access structure $\mathbb{A}_1$, Alice can invoke $\mathbf{Tran}_{\mathsf{V}}$ to gain a full signature σ_B of Bob's. Then Alice invokes $\mathbf{Tran}_{\mathsf{S}}$ to generate a credential-protected package π_A on message m_1 with respect to access structures $\mathbb{A}_2$. Alice sends π_A to Bob, who can invoke $\mathbf{Tran}_{\mathsf{V}}$ to gain Alice's full signature σ_A if satisfying the access structure $\mathbb{A}_2$.

Note that in the normal cases, the exchange will finish and both Alice and Bob can gain the full signature of the other's. In the case Alice refuses to send her credential-protected package π_A in the third step, or that credential-protected package is created with respect to improper access structures, Bob can approach the arbitrator for assistance. Specifically, he approaches the arbitrator and sends a resolution request generated by invoking $\mathbf{Res}_{\mathsf{User}}$. To make a resolution, the arbitrator invokes $\mathbf{Res}_{\mathsf{TTP}}$ to generate two credential-protected packages $\tilde{\pi}_1$ and $\tilde{\pi}_2$. $\tilde{\pi}_1$ is sent to Bob, and $\tilde{\pi}_2$ is sent to Alice. Thus both Alice and Bob can invoke $\mathbf{Tran}_{\mathsf{V}}$ to gain the other's full signature if satisfying the expectant attributes.

3.2. Security model

We modify the traditional OFE model to make it suitable in the attribute-based setting.

- **Security Against Signers**: This property guarantees that a transaction promise generated by a signer can always be resolved to a full signature of the signer's if the arbitrator honestly makes a resolution and the verifier satisfies the expectant attributes. Formally, we consider

the following experiment, in which the adversary $\mathcal{A}$ models a dishonest signer that can even have access to the credential secret key and the verifier's secret key.

Experiment $\mathtt{SAS}$:

$$
\begin{aligned}
(\mathsf{PM}, \mathsf{CK}) &\leftarrow \mathbf{PMGen}(1^k, U) \\
(\mathsf{ASK}, \mathsf{APK}) &\leftarrow \mathbf{Setup}^{\mathsf{TTP}}(1^k) \\
(\mathsf{SK}_B, \mathsf{PK}_B) &\leftarrow \mathbf{Setup}^{\mathsf{User}}(1^k) \\
(m_1^*, m_2^*, \omega^*, \mathsf{PK}_A, \mathbb{A}_1, \mathbb{A}_2, S) &\leftarrow \mathcal{A}^{O_{\mathbf{Res}}}(\mathsf{PM}, \mathsf{CK}, \mathsf{APK}, \mathsf{SK}_B, \mathsf{PK}_B) \\
\mathsf{CD}_S &\leftarrow \mathbf{Setup}^{\mathsf{Cred}}(\mathsf{CK}, S) \\
\mathsf{request}^* &\leftarrow \mathbf{Res}_{\mathsf{User}}(m_1^*, m_2^*, \omega^*, \mathsf{SK}_B, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2) \\
(\pi_1^*, \pi_2^*) &\leftarrow \mathbf{Res}_{\mathsf{TTP}}(m_1^*, \mathsf{PK}_A, \mathbb{A}_2, \mathsf{request}^*, \mathsf{ASK}) \\
\sigma^* &\leftarrow \mathbf{Tran}_{\mathsf{V}}(m_1^*, \pi_1^*, \mathsf{PK}_A, \mathsf{APK}, \mathsf{PM}, \mathsf{CD}_S) \\
\text{success of } \mathcal{A} &:= [\mathbf{TPVer}(m_1^*, m_2^*, \omega^*, \mathsf{PK}_A, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2) = \top \\
&\quad \wedge\ \mathbf{Ver}(m_1^*, \sigma^*, \mathsf{PK}_A, \mathsf{APK}) = \bot]
\end{aligned}
$$

where S is a set of attributes that satisfies the access structure $\mathbb{A}_2$, the resolution for arbitrator oracle $O_{\mathbf{Res}}$ takes as input $(m, \mathsf{PK}_i, \mathbb{A}, \mathsf{request})$, and outputs $\mathbf{Res}_{\mathsf{TTP}}(m, \mathsf{PK}_i, \mathbb{A}, \mathsf{request}, \mathsf{ASK})$. In this experiment, the adversary can arbitrarily choose a public key PK_i, and it may not know the corresponding private key of PK_i.

- **Security Against Verifiers**: This property captures two cases: the first case is that no verifier, even given the credential secret key, should be able to generate a full signature of the signer's while not possessing a credential-protected package. The second case is that the verifier that does not possess the expectant attributes should not be able to generate a full signature even if given a corresponding credential-protected package. Formally, we require that no PPT adversary $\mathcal{A}$, who models a dishonest verifier, succeeds with non-negligible probability in either of the following two experiments:

Experiment SAV1:

$$\begin{aligned}
(\mathsf{PM}, \mathsf{CK}) &\leftarrow \mathbf{PMGen}(1^k, U) \\
(\mathsf{ASK}, \mathsf{APK}) &\leftarrow \mathbf{Setup}^{\mathsf{TTP}}(1^k) \\
(\mathsf{SK}_A, \mathsf{PK}_A) &\leftarrow \mathbf{Setup}^{\mathsf{User}}(1^k, \mathsf{APK}) \\
(m^*, \sigma^*) &\leftarrow \mathcal{A}^{O_{\mathbf{Tran_P}}, O_{\mathbf{Res}}}(\mathsf{PM}, \mathsf{CK}, \mathsf{APK}, \mathsf{PK}_A) \\
\text{success of } \mathcal{A} &:= [\mathbf{Ver}(m^*, \sigma^*, \mathsf{PK}_A, \mathsf{APK}) = \top \\
&\quad \wedge (m^*, \mathsf{PK}_A, \cdot, \cdot) \notin Query(\mathcal{A}, O_{\mathbf{Res}})]
\end{aligned}$$

where oracle $O_{\mathbf{Res}}$ is described in the previous experiment, oracle $O_{\mathbf{Tran_P}}$ takes as input $(m_1, m_2, \mathbb{A}, \mathbb{A}')$ and outputs $\mathbf{Tran_P}(m_1, m_2, \mathsf{SK}_A, \mathsf{APK}, \mathsf{PM}, \mathbb{A}, \mathbb{A}')$, and $Query(\mathcal{A}, O_{\mathbf{Res}})$ is the set of queries $\mathcal{A}$ issued to the resolution oracle. Note that there is no need to provide $\mathcal{A}$ with access to the transaction for signer oracle $O_{\mathbf{Trans_S}}$, as resolution ambiguity property guarantees that its functionality could be achieved by executing $O_{\mathbf{Tran_P}}$ and $O_{\mathbf{Res}}$.

Experiment SAV2:

$$\begin{aligned}
(\mathsf{PM}, \mathsf{CK}) &\leftarrow \mathbf{PMGen}(1^k, U) \\
(\mathsf{ASK}, \mathsf{APK}) &\leftarrow \mathbf{Setup}^{\mathsf{TTP}}(1^k) \\
(\mathsf{SK}_A, \mathsf{PK}_A) &\leftarrow \mathbf{Setup}^{\mathsf{User}}(1^k, \mathsf{APK}) \\
(m^*, \sigma^*) &\leftarrow \mathcal{A}^{O_{\mathbf{Setup}^{\mathsf{Cred}}}, O_{\mathbf{Tran_P}}, O_{\mathbf{Res}}}(\mathsf{PM}, \mathsf{APK}, \mathsf{PK}_A) \\
\text{success of } \mathcal{A} &:= [\mathbf{Ver}(m^*, \sigma^*, \mathsf{PK}_A, \mathsf{APK}) = \top \\
&\quad \wedge \text{ for any } S \in Query(\mathcal{A}, O_{\mathbf{Setup}^{\mathsf{Cred}}}), \\
&\quad\quad \text{for any } (m^*, \mathsf{PK}_A, \mathbb{A}, \cdot) \in Query(\mathcal{A}, O_{\mathbf{Res}}), \\
&\quad\quad S \text{ does not satisfy } \mathbb{A}]
\end{aligned}$$

where oracles $O_{\mathbf{Tran_P}}$, $O_{\mathbf{Res}}$ and $Query(\mathcal{A}, O_{\mathbf{Res}})$ are described in the previous experiment, oracle $O_{\mathbf{Setup}^{\mathsf{Cred}}}$ takes as input a set of attributes S, and outputs a credential CD_S, and $Query(\mathcal{A}, O_{\mathbf{Setup}^{\mathsf{Cred}}})$ is the set of queries $\mathcal{A}$ issued to the oracle $O_{\mathbf{Setup}^{\mathsf{Cred}}}$.

- **Security Against the Arbitrator**: This property guarantees that even when given the credential secret key, the arbitrator should not be able to create a valid full signature unless it has seen a corresponding

transaction promise. Formally, we consider the following experiment, in which the two-stage adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ models the dishonest arbitrator that may even have access the master key.

Experiment $\mathtt{SAA}$:

$$\begin{aligned}
(\mathsf{PM}, \mathsf{CK}) &\leftarrow \mathbf{PMGen}(1^k, U) \\
(\mathsf{APK}, \mathsf{ASK}^*) &\leftarrow \mathcal{A}_1(\mathsf{PM}, \mathsf{MK}) \\
(\mathsf{SK}_A, \mathsf{PK}_A) &\leftarrow \mathbf{Setup}^{\mathsf{User}}(1^k, \mathsf{APK}) \\
(m^*, \sigma^*) &\leftarrow \mathcal{A}_2^{O_{\mathbf{Tran_P}}}(\mathsf{PM}, \mathsf{CK}, \mathsf{ASK}^*, \mathsf{APK}, \mathsf{PK}_A) \\
\text{success of } \mathcal{A} &:= [\mathbf{Ver}(m^*, \sigma^*, \mathsf{PK}_A, \mathsf{APK}) = \top \\
&\quad \wedge (m^*, \cdot, \cdot, \cdot) \notin \mathit{Query}(\mathcal{A}, O_{\mathbf{Tran_P}})]
\end{aligned}$$

where the partial signing oracle $O_{\mathbf{Tran_P}}$ is described in the previous experiment, ASK^* is $\mathcal{A}$'s state information, which might not be the corresponding secret key of APK, and $\mathit{Query}(\mathcal{A}, O_{\mathbf{Tran_P}})$ is the set of queries $\mathcal{A}$ issued to oracle $O_{\mathbf{Tran_P}}$.

Definition 5. *An attribute-based optimistic fair exchange scheme is said to be secure in the multi-user setting and chosen-key model if there is no PPT adversary that wins any of the experiments above with non-negligible probability.*

4. Construction

In this section, we give a generic construction of ABOFE that is secure in our defined model.

Let $\mathcal{E} := (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{KeyGen}, \mathsf{Decrypt})$ be a ciphertext-policy attribute-based encryption scheme and $\mathsf{OFE} = (\mathsf{Setup}^{\mathsf{TTP}}, \mathsf{Setup}^{\mathsf{User}}, \mathsf{PSig}, \mathsf{PVer}, \mathsf{Sig}, \mathsf{Ver}, \mathsf{Res})$ be a conventional optimistic fair exchange scheme. Below are the details of of our generic construction of an attribute-based optimistic fair exchange scheme.

- **PMGen**: On input 1^k and attribute universe description U, this algorithm runs $\mathcal{E}.\mathsf{Setup}(1^k, U) \rightarrow (\mathrm{PM}, \mathrm{MK})$. The public parameter and credential secret key are set as $\mathsf{PM} := \mathrm{PM}$ and $\mathsf{CK} := \mathrm{MK}$, respectively.

- **Setup**$^{\mathsf{TTP}}$: On input 1^k, this algorithm runs $\mathsf{OFE.Setup}^{\mathsf{TTP}}(1^k) \rightarrow (ASK, APK)$. The arbitrator's secret and public key pair is set as $\mathsf{ASK} := ASK$ and $\mathsf{APK} := APK$.

- **Setup**$^{\mathsf{User}}$: This algorithm runs $\mathsf{OFE.Setup}^{\mathsf{User}}(1^k, APK) \rightarrow (SK_i, PK_i)$. The user's secret and public key pair is set as $\mathsf{SK}_i := SK_i, \mathsf{PK}_i := PK_i$.

- **Setup**$^{\mathsf{Cred}}$: On input the credential secret key CK and a set of attributes S, this algorithm runs $\mathcal{E}.\mathsf{KeyGen}\ (\mathsf{MK}, S) \rightarrow \mathrm{SK}$. The credential is set as $\mathsf{CD}_S := \mathrm{SK}$.

- **Tran**$_{\mathsf{P}}$: Taking as input $(m_1, m_2, \mathsf{SK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2)$, user U_i runs $\mathsf{OFE.PSig}(m_1, \mathsf{SK}_i,\ \mathsf{APK}) \rightarrow \sigma_P$, and generates a non-repudiation information θ about the transaction[2] with respect to the tuple $(\sigma_P, m_1, m_2, \mathbb{A}_1, \mathbb{A}_2)$. The transaction promise ω is set as (σ_P, θ)[3].

- **TPVer**: Taking as input $(m_1, m_2, \omega, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2)$, it outputs $\mathsf{OFE.PVer}(m_1, \sigma_P,\ \mathsf{PK}_i,\ \mathsf{APK})$.

- **Tran**$_{\mathsf{S}}$: Taking as input $(m, \mathsf{SK}_i,\ \mathsf{APK}, \mathsf{PM}, \mathbb{A})$, user U_i runs $\mathsf{OFE.Sig}(m, \mathsf{SK}_i, \mathsf{APK}) \rightarrow s$, and then encrypts s under the access structure $\mathbb{A}$, i.e., $\mathcal{E}.\mathsf{Encrypt}(\mathsf{PM}, s, \mathbb{A}) \rightarrow \mathrm{CT}$. The credential-protected package is set as $\pi := \mathrm{CT}$.

- **Tran**$_{\mathsf{V}}$: Taking as input $(m, \pi, \mathsf{PK}_i,\ \mathsf{APK}, \mathsf{PM}, \mathsf{CD}_S)$, the verifier that possesses a set of attributes S that satisfies the access structure $\mathbb{A}$ will be able to convert the credential-protected package to a full signature. The verifier outputs $\sigma \leftarrow \mathcal{E}.\mathsf{Decrypt}(\mathsf{PM}, \pi, \mathsf{CD}_S)$.

[2]This is used to identify the transaction that is going on and will be necessary when the arbitrator makes a resolution. Typically this can be achieved by signing $\sigma_P||m_1||m_2||\mathbb{A}_1||\mathbb{A}_2$ using an independent signature key pair.

[3]In most research of optimistic fair exchange, it is normally assumed that only the full signatures are useful to the exchanging parties. Even if Bob holds a partial signature from Alice, Bob can not acquire Alice's service, since only the full signature represents the commitment of a service. Thus, for simplicity and clarity, we send the partial signature σ_P in clear. In practice, if we would like to achieve the property that Bob cannot gain Alice's partial signature unless he satisfies some attributes, then Alice can encrypt σ_P under a proper access structure.

- **Ver**: Taking as input $(m, \sigma, \mathsf{PK}_i, \mathsf{APK})$, it outputs $\mathsf{OFE.Ver}(m, \sigma, \mathsf{PK}_i, \mathsf{APK})$.

- $\mathbf{Res}_{\mathsf{User}}$: Taking as input $(m_1, m_2, \omega, \mathsf{SK}_j, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2)$, it runs $\mathsf{OFE.Sig}(m_2, \mathsf{SK}_j\ \mathsf{APK}) \rightarrow \sigma_2$. The resolution request is set as $\mathsf{request} := (m_2, \omega, \mathsf{PK}_j, \sigma_2, \mathbb{A}_1)$.

- $\mathbf{Res}_{\mathsf{TTP}}$: Taking as input $(m, \mathsf{PK}_i, \mathbb{A}, \mathsf{request}, \mathsf{ASK})$, the arbitrator firstly parses $\mathsf{request}$ as $(m_2, \omega, \mathsf{PK}_j, \sigma_2, \mathbb{A}_1)$, and checks whether $\mathbf{TPVer}(m_1, m_2, \omega, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2) = \top$ and whether $\mathbf{Ver}(m_2, \sigma_2, \mathsf{PK}_j, \mathsf{APK}) = \top$. If either does not hold, it returns $\perp$. Otherwise, the arbitrator parses ω as (σ_P, θ), computes $\sigma_1 \leftarrow \mathsf{OFE.Res}(m, \sigma_P, \mathsf{ASK}, \mathsf{PK}_i)$ and then encrypts σ_1 under the access structure $\mathbb{A}_2$, i.e., $\mathcal{E}.\mathsf{Encrypt}(\mathsf{PM}, \sigma_1, \mathbb{A}_2) \rightarrow \mathrm{CT}_1$. The arbitrator also encrypts σ_2 under the access structure $\mathbb{A}_1$, i.e., $\mathcal{E}.\mathsf{Encrypt}(\mathsf{PM}, \sigma_2, \mathbb{A}_1) \rightarrow \mathrm{CT}_2$. The credential-protected packages are set as $\pi_1 := \mathrm{CT}_1$ and $\pi_2 := \mathrm{CT}_2$.

4.1. Security Analysis

Regarding the security of our generic construction of ABOFE, we have the following theorem.

Theorem 1. *Our generic construction of ABOFE is secure in the multi-user setting and chosen-key model if the optimistic fair exchange scheme* OFE *is secure in the multi-user setting and chosen-key model and the ciphertext-policy attribute-based encryption scheme* $\mathcal{E}$ *is fully secure.*

We prove Theorem 1 by the following three lemmas.

Lemma 1 (Security against Signers). *The generic construction is secure against signers if* $\mathsf{OFE} = (\mathsf{Setup}^{\mathsf{TTP}}, \mathsf{Setup}^{\mathsf{User}}, \mathsf{PSig}, \mathsf{PVer}, \mathsf{Sig}, \mathsf{Ver}, \mathsf{Res})$ *is secure against signers.*

Proof. To show security against signers, we convert any adversary $\mathcal{A}$ that wins the experiment $\mathtt{SAS}$ into an adversary $\mathcal{A}'$ that breaks the security against signers of OFE. Recall that $\mathcal{A}'$ gets APK as input and has access to oracle O'_{Res}. $\mathcal{A}'$ runs $\mathbf{PMGen}(1^k, U) \rightarrow (\mathsf{PM}, \mathsf{CK})$ and invokes $\mathcal{A}$ on input PM, CK and $\mathsf{APK} := APK$.

Given a resolution for arbitrator query $(m, \mathsf{PK}_i, \mathbb{A}, \mathsf{request})$ to $O_{\mathbf{Res}}$, $\mathcal{A}'$ firstly parses $\mathsf{request}$ as $(m_2, \omega, \mathsf{PK}_j, \sigma_2, \mathbb{A}')$ and checks whether $\mathbf{TPVer}$

$(m_1, m_2, \omega, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A}', \mathbb{A}) = \top$ and whether $\mathbf{Ver}(m_2, \sigma_2, \mathsf{PK}_j, \mathsf{APK}) = \top$. If either does not hold, it returns $\perp$. Otherwise, $\mathcal{A}'$ parses ω as (σ_P, θ), and makes a query $(m, \sigma_P, \mathsf{PK}_i)$ to its own oracle O'_{Res}. Denote the answer from O'_{Res} is σ_1. $\mathcal{A}'$ encrypts σ_1 under the access structure $\mathbb{A}$, i.e., $\mathcal{E}.\mathsf{Encrypt}(\mathsf{PM}, \sigma_1, \mathbb{A}) \rightarrow \mathrm{CT}_1$. $\mathcal{A}'$ also encrypts σ_2 under the access structure $\mathbb{A}'$, i.e., $\mathcal{E}.\mathsf{Encrypt}(\mathsf{PM}, \sigma_2, \mathbb{A}') \rightarrow \mathrm{CT}_2$. The credential-protected packages are set as $\pi_1 := \mathrm{CT}_1$ and $\pi_2 := \mathrm{CT}_2$. $\mathcal{A}'$ returns (π_1, π_2) to $\mathcal{A}$.

It can be seen that the resolution for arbitrator oracle $O_{\mathbf{Res}}$ is perfectly simulated by $\mathcal{A}'$. Finally, $\mathcal{A}$ outputs a tuple $(m_1^*, m_2^*, (\sigma_P^*, \theta^*), \mathsf{PK}_A, \mathbb{A}_1, \mathbb{A}_2, S)$ where the set of attributes S satisfies the access structure $\mathbb{A}_2$. $\mathcal{A}'$ outputs $(m_1^*, \sigma_P^*, \mathsf{PK}_A)$.

Denote (π_1^*, π_2^*) as the output of $\mathbf{Res}_{\mathsf{TTP}}(m_1^*, \mathsf{PK}_A, \mathbb{A}_2, \mathsf{Request}^*, \mathsf{ASK})$, and σ^* as the output of $\mathbf{Tran}_{\mathsf{V}}$ $(m_1^*, \pi_1^*, \mathsf{PK}_A, \mathsf{APK}, \mathsf{PM}, \mathsf{CD}_S)$ where $\mathsf{CD}_S = \mathbf{Setup}^{\mathsf{Cred}}(\mathsf{CK}, S)$. Due to the fact that the set of attributes S satisfies the access structure $\mathbb{A}_2$, σ^* is in fact also the output of $\mathsf{OFE}.\mathsf{Res}(m_1^*, \sigma_P^*, \mathsf{ASK}, \mathsf{PK}_A)$. Since $\mathbf{TPVer}(m_1^*, m_2^*, (\sigma_P^*, \theta^*), \mathsf{PK}_A, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2) = \top$ but $\mathbf{Ver}(m_1^*, \sigma^*, \mathsf{PK}_A, \mathsf{APK}) = \perp$, it means $\mathsf{OFE}.\mathsf{PVer}$ $(m_1^*, \sigma_P^*, PK_A, APK) = \top$ but $\mathsf{OFE}.\mathsf{Ver}$ $(m_1^*, \sigma^*, PK_A, APK) = \perp$. Thus if $\mathcal{A}$ succeeds in the experiment, $\mathcal{A}'$ also succeeds with the same probability in breaking the security against signers of OFE. □

Lemma 2 (Security against Verifiers). *The generic construction is secure against verifiers if* $\mathsf{OFE} = (\mathsf{Setup}^{\mathsf{TTP}}, \mathsf{Setup}^{\mathsf{User}}, \mathsf{PSig}, \mathsf{PVer}, \mathsf{Sig}, \mathsf{Ver}, \mathsf{Res})$ *is secure against verifiers and the ciphertext-policy attribute-based encryption* $\mathcal{E} := (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{KeyGen}, \mathsf{Decrypt})$ *is fully secure.*

Proof. To show security against verifiers, we consider the experiments `SAV1` and `SAV2`, respectively. We firstly convert any adversary $\mathcal{A}$ that wins the experiment `SAV1` into an adversary $\mathcal{A}'$ that breaks the security against verifiers of OFE.

Recall that $\mathcal{A}'$ gets (APK, PK_A) as input and has access to oracles O'_{PSig} and O'_{Res}. $\mathcal{A}'$ runs $\mathbf{PMGen}(1^k, U) \rightarrow (\mathsf{PM}, \mathsf{MK})$ and invokes $\mathcal{A}$ on input PM, CK, $\mathsf{APK} := APK$ and $\mathsf{PK}_A := PK_A$.

Given a transaction promise query $(m_1, m_2, \mathbb{A}, \mathbb{A}')$ to oracle $O_{\mathbf{Tran}_{\mathsf{P}}}$, $\mathcal{A}'$ makes a query m to its own oracle O'_{PSig}. Denote the answer from O'_{PSig} is σ_P. $\mathcal{A}'$ returns (σ_P, θ) to $\mathcal{A}$ where θ is a non-repudiation information about the transaction with respect to the tuple $(\sigma_P, m_1, m_2, \mathbb{A}_1, \mathbb{A}_2)$.

Given a resolution for arbitrator query $(m, \mathsf{PK}_i, \mathbb{A}, \mathsf{request})$ to $O_{\mathbf{Res}}$, $\mathcal{A}'$ firstly parses $\mathsf{request}$ as $(m_2, \omega, \mathsf{PK}_j, \sigma_2, \mathbb{A}')$ and checks whether $\mathbf{TPVer}$

$(m_1, m_2, \omega, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, \mathbb{A}', \mathbb{A}) = \top$ and whether $\mathbf{Ver}(m_2, \sigma_2, \mathsf{PK}_j, \mathsf{APK}) = \top$. If either does not hold, it returns $\perp$. Otherwise, $\mathcal{A}'$ parses ω as (σ_P, θ), and makes a query $(m, \sigma_P, \mathsf{PK}_i)$ to its own oracle O'_{Res}. Denote the answer from O'_{Res} is σ_1. $\mathcal{A}'$ encrypts σ_1 under the access structure $\mathbb{A}$, i.e., $\mathcal{E}.\mathsf{Encrypt}(\mathsf{PM}, \sigma_1, \mathbb{A}) \rightarrow \mathrm{CT}_1$. $\mathcal{A}'$ also encrypts σ_2 under the access structure $\mathbb{A}'$, i.e., $\mathcal{E}.\mathsf{Encrypt}(\mathsf{PM}, \sigma_2, \mathbb{A}') \rightarrow \mathrm{CT}_2$. The credential-protected packages are set as $\pi_1 := \mathrm{CT}_1$ and $\pi_2 := \mathrm{CT}_2$. $\mathcal{A}'$ returns (π_1, π_2) to $\mathcal{A}$.

It can be seen that the oracles $O_{\mathbf{PSig}}$ and $O_{\mathbf{Res}}$ are perfectly simulated by $\mathcal{A}'$. Finally, $\mathcal{A}$ outputs a tuple (m^*, σ^*) such that $\mathbf{Ver}(m^*, \sigma^*, \mathsf{PK}_A, \mathsf{APK}) = \top$. This means $\mathsf{OFE.Ver}(m^*, \sigma^*, PK_A, APK) = \top$. $\mathcal{A}'$ outputs (m^*, σ^*). Since $\mathcal{A}$ is prohibited from making a query $(m^*, \mathsf{PK}_A, \cdot, \cdot)$ to oracle $O_{\mathbf{Res}}$, $\mathcal{A}'$ has never made a query with respect to a tuple $(m^*, \cdot, \mathsf{PK}_A)$ to its own oracle O'_{Res}. If $\mathcal{A}$ succeeds in the experiment $\mathtt{SAV1}$, $\mathcal{A}'$ also succeeds in breaking the security against verifiers of OFE. Thus $\mathcal{A}'$'s advantage is also non-negligible.

Next, we consider the experiment $\mathtt{SAV2}$. We convert any adversary $\mathcal{A}$ that wins the experiment $\mathtt{SAV2}$ into an algorithm $\mathcal{B}$ that breaks the full security of the encryption scheme $\mathcal{E}$.

Recall that $\mathcal{B}$ gets PM as input and can query its own challenger for private keys corresponding to a sequence of sets of attributes. $\mathcal{B}$ runs $\mathbf{Setup}^{\mathsf{TTP}}(1^k) \rightarrow (\mathsf{ASK}, \mathsf{APK})$ and $\mathbf{Setup}^{\mathsf{User}}(1^k, \mathsf{APK}) \rightarrow (\mathsf{SK}_A, \mathsf{PK}_A)$, and invokes $\mathcal{A}$ as a subroutine by forwarding $\mathsf{PM} := \mathrm{PM}$, APK and PK_A.

When $\mathcal{A}$ makes a query S to oracle $O_{\mathbf{Setup}^{\mathsf{Cred}}}$, $\mathcal{B}$ supplies S to its own challenger and forwards the reply to $\mathcal{A}$. When $\mathcal{A}$ makes a transaction promise query $(m_1, m_2, \mathbb{A}, \mathbb{A}')$ to oracle $O_{\mathbf{Tran_P}}$, $\mathcal{B}$ runs $\mathbf{Tran}_{\mathsf{P}}(m_1, m_2, \mathsf{SK}_A, \mathsf{APK}, \mathsf{PM}, \mathbb{A}_1, \mathbb{A}_2)\omega$ and send ω to $\mathcal{A}$ as the reply.

Suppose $\mathcal{A}$ makes q valid resolution for arbitrator queries to $O_{\mathbf{Res}}$. $\mathcal{B}$ chooses a random value $z \in \{1, \cdots, q\}$. When $\mathcal{A}$ makes a j-th resolution query $(m_j, , \mathsf{PK}_j, \mathbb{A}_j, \mathsf{request}_j)$ to $O_{\mathbf{Res}}$, if $j \neq z$, $\mathcal{B}$ runs $\mathbf{Res}_{\mathsf{TTP}}(m_z, \mathsf{PK}_j, \mathbb{A}_j, \mathsf{request}_j, \mathsf{ASK})$ and sends the outputs to $\mathcal{A}$ as the reply. When $j = z$ and $\mathsf{PK}_z \neq \mathsf{PK}_A$, $\mathcal{B}$ aborts and returns failure. When $j = z$ and $\mathsf{PK}_z = \mathsf{PK}_A$, $\mathcal{B}$ parses $\mathsf{request}_z$ as $(m_2, \omega_j, \mathsf{PK}_i, \sigma_2, \mathbb{A}'_z)$ and parses ω_z as (σ_{Pz}, θ_z). $\mathcal{B}$ runs $\mathsf{OFE.Res}(m_z, \sigma_{Pz}, \mathsf{ASK}, \mathsf{PK}_A) \rightarrow \sigma_z$, randomly chooses M_1 that is of the same bit length with σ_z, and sends $M_0 := \sigma_z$, M_1 and the access structure $\mathbb{A}_z$ to its own challenger. Denote the challenge ciphertext is CT. $\mathcal{B}$ also encrypts σ_2 under the access structure $\mathbb{A}'_z$, i.e., $\mathcal{E}.\mathsf{Encrypt}(\mathsf{PM}, \sigma_2, \mathbb{A}') \rightarrow \mathrm{CT}'$. The credential-protected packages are set as $\pi_1 := \mathrm{CT}$ and $\pi_2 := \mathrm{CT}'$. $\mathcal{A}'$ returns (π_1, π_2) to $\mathcal{A}$ as the reply to the z-th resolution for arbitrator query.

Finally $\mathcal{A}$ either outputs failure or wins by outputting a tuple (m^*, σ^*) such that $\mathbf{Ver}(m^*, \sigma^*, \mathsf{PK}_A, \mathsf{APK}) = \top$. Note that when $\mathcal{A}$ wins, $\mathcal{A}$ must have made a resolution query $(m^*, \mathsf{PK}_A, \cdot, \cdot)$ to $O_{\mathbf{Res}}$. Otherwise the analysis of this type of attack is covered in the experiment `SAV1` discussed above. If $m^* = m_z$, $\mathsf{PK}_A = \mathsf{PK}_z$, and $\mathbb{A}_2 = \mathbb{A}'_z$, then $\mathcal{B}$ outputs 0. Otherwise, $\mathcal{B}$ outputs a random bit.

If the challenge ciphertext CT is the encryption of σ_z (i.e., $b = 0$), CT is a valid attribute-related signature and the distribution of B's view in the simulated environment is identical with that in the real attack environment. If the challenge ciphertext CT is the encryption of M_1 (i.e., $b = 1$), CT has no information on the full signature of a message m_z and B's chance of forging a valid full signature on message m_z (with respect to PK_A) is negligible. Thus if $\mathcal{A}$ in the real experiment `SAV2` wins with non-negligible probability, then $\mathcal{B}$ wins with non-negligible probability in breaking the full security of the encryption scheme $\mathcal{E}$. □

Lemma 3 (Security against the Arbitrator). *The generic construction is secure against the arbitrator if* $\mathsf{OFE} = (\mathsf{Setup}^{\mathsf{TTP}}, \mathsf{Setup}^{\mathsf{User}}, \mathsf{PSig}, \mathsf{PVer}, \mathsf{Sig}, \mathsf{Ver}, \mathsf{Res})$ *is secure against the arbitrator.*

Proof. To show security against the arbitrator, we convert any adversary $\mathcal{A}$ that wins the experiment `SAA` into an adversary $\mathcal{A}'$ that breaks the security against the arbitrator of OFE. $\mathcal{A}$ firstly chooses a public adjudication key APK and outputs it, keeps a corresponding secret state information ASK^* private. $\mathcal{A}'$ sets $APK := \mathsf{APK}$, gets PK_A as input, and has access to oracles O'_{PSig}. $\mathcal{A}'$ runs $\mathbf{PMGen}(1^k, U) \rightarrow (\mathsf{PM}, \mathsf{CK})$ and forwards PM, CK and $\mathsf{PK}_A := PK_A$ to $\mathcal{A}$.

Given a transaction promise query $(m_1, m_2, \mathbb{A}, \mathbb{A}')$ to oracle $O_{\mathbf{Tran_P}}$, $\mathcal{A}'$ makes a query m to its own oracle O'_{PSig}. Denote the answer from O'_{PSig} is σ_P. $\mathcal{A}'$ returns (σ_P, θ) to $\mathcal{A}$ where θ is a non-repudiation information about the transaction with respect to the tuple $(\sigma_P, m_1, m_2, \mathbb{A}_1, \mathbb{A}_2)$.

It can be seen that the oracle $O_{\mathbf{Tran_P}}$ is perfectly simulated by $\mathcal{A}'$. Finally, $\mathcal{A}$ outputs a tuple (m^*, σ^*) such that $\mathbf{Ver}$ $(m^*, \sigma^*, \mathsf{PK}_A, \mathsf{APK} = \top$. This means $\mathsf{OFE.Ver}$ $(m^*, \sigma^*, PK_A, APK) = \top$. $\mathcal{A}'$ outputs (m^*, σ^*). Since $\mathcal{A}$ is prohibited from making a query $(m^*, \cdot, \cdot, \cdot)$ to oracle $O_{\mathbf{Tran_P}}$, $\mathcal{A}'$ has never made a query message m^* to its own oracle O'_{PSig}. If $\mathcal{A}$ succeeds in the experiment `SAV2`, $\mathcal{A}'$ also succeeds with the same probability in breaking the security against the arbitrator of OFE. □

5. Instantiation

In the following, we provide an instantiation to demonstrate the flexibility of our generic construction. It was shown in [18] that OFE secure in the multi-user setting and chosen key model can be constructed from conventional signatures and ring signatures. More specifically, in this paradigm, the partial signature in OFE is a conventional signature, and the full signature in OFE is the partial signature, together with a two party ring signature generated between the signer and the arbitrator. The authors [18] suggest a concrete OFE scheme can be built on waters signature scheme [25] in group of composite order and Shacham-Waters' ring signature scheme [26] so that they share the same set of system parameters. For the CP-ABE scheme, we employ the one proposed by Lewko and Waters in [23], which is known to be fully secure. Note that there is a global setup process before execution of the scheme due to the requirement of having such a setup process of Shacham-Waters' ring signature.

Global Setup : On input 1^k where k is a security parameter, the setup algorithm generates a multiplicative cyclic group G of composite order n_1 where $n_1 = pq$ and a bilinear pairing $e : G \times G \rightarrow G_T$ where G_T is a multiplicative group of order n_1. Let G_p and G_q be the cyclic order-p and order-q subgroups of G, respectively. Let g be a generator of G and h be a generator of G_q. It then chooses random exponents $a, b \in \mathbb{Z}_{n_1}$ and sets

$$A := g^a, B := g^b, \hat{A} := h^a.$$

Let $H_0 : \{0,1\}^* \rightarrow \{0,1\}^k$ be a collision-resistant hash function. The setup algorithm picks Waters hash generators

$$u', u_1, \cdots, u_k \leftarrow G.$$

The common reference string is set as $(n_1, A, B, \hat{A}, u', u_1, \cdots, u_k, H_0)$.

After the global setup is finished, the algorithms in ABOFE can be executed as follows.

- **PMGen**: On input 1^k where k is a security parameter, the setup algorithm outputs a multiplicative cyclic group $\hat{G}$ of order n_2 where $n_2 = p_1p_2p_3$ (3 distinct primes) and a bilinear pairing $\hat{e} : \hat{G} \times \hat{G} \rightarrow \hat{G}_T$ where $\hat{G}_T$ is a multiplicative group of order n_2. Let $\hat{G}_{p_i}$ denote the

subgroup of order p_i in $\hat{G}$. Let g_i be a generator of $\hat{G}_{p_i}$ and $H_1 : \hat{G}_T \to \{0,1\}^l$ be a collision-resistant hash function. It then chooses random exponents $\alpha, c, \gamma \in \mathbb{Z}_n$, and for each attribute $i \in U$, it chooses a random group element $h_i \in \hat{G}_{p_1}$. Let SE be a symmetric encryption scheme whose symmetric key space is $\{0,1\}^l$. The public parameters PM are set as

$$n_2, g_1, \hat{A}, g_1^c, g_1^\gamma, e(g_1,g_1)^\alpha, H_1, h_1, \cdots, h_{|U|}, \mathsf{SE}.$$

The master key are set as $\mathsf{MK} := (g_1^\alpha, g_3)$.

- **Setup**$^{\mathsf{TTP}}$: The arbitrator chooses a random exponent $y \in \mathbb{Z}_{n_1}$, and sets $\mathsf{APK} := g^y \in G$ and $sk := A^y \in G$.

- **Setup**$^{\mathsf{User}}$: Each user U_i randomly chooses exponents $x_1, x_3 \in \mathbb{Z}_{n_1}$ and computes $\bar{g}_1 = g^{x_1}$ and $\bar{g}_3 = g^{x_3}$. Besides, user U_i randomly chooses $\bar{g}_2, \bar{u}', \bar{u}_1, \cdots, \bar{u}_k \leftarrow G$ and sets $\mathsf{PK}_i := (\bar{g}_1, \bar{g}_2, \bar{g}_3, \bar{u}', \bar{u}_1, \cdots, \bar{u}_k)$ and $\mathsf{SK} := (\bar{g}_2^{x_1}, A^{x_3})$.

- **Setup**$^{\mathsf{Cred}}$: On input the credential secret key CK and a set of attributes S, the algorithm chooses random exponents $t, u \in \mathbb{Z}_{n_2}$, and random elements $R, R', R'', \{R_i\}_{i\in S} \in G_{p_3}$ (this can be done by raising a generator of G_{p_3} to random exponents modulo n_2). The credential is:

$$S,\ K = g_1^\alpha g_1^{ct} g_1^{\gamma u} R,\ K' = g_1^u R',\ K'' = g^t R'',\ K_i = h_i^t R_i \text{ for } i \in S.$$

- **Tran**$_{\mathsf{P}}$: **Tran**$_{\mathsf{P}}(m_A, m_B, \mathsf{SK}_i, \mathsf{APK}, \mathsf{PM}, (M,\rho), (M',\rho'))$, where M is an $L \times N$ matrix and ρ a map from each row M_j of M to an attribute $\rho(j)$, and M' is an $L' \times N'$ matrix and ρ' a map from each row M'_j of M' to an attribute $\rho'(j)$, does as follows:

  1. Compute $(m_1, \cdots, m_k) \leftarrow H_0(m_A)$,
  2. Choose a random exponent $r \in \mathbb{Z}_{n_1}$ and computes

$$S_1 = \bar{g}_2^{x_1} \cdot (\bar{u}' \prod_{i=1}^{k} \bar{u}_i^{m_i})^r, \quad \text{and} \quad S_2 = g^r.$$

  3. σ_P is set as (S_1, S_2).

4. Generate a non-repudiation information θ about the transaction with respect to the tuple $(\sigma_P, m_A,\ m_B, (M, \rho), (M', \rho'))$. The transaction promise ω is set as (σ_P, θ).

- **TPVer**: $\mathbf{TPVer}(m_A, m_B, \omega, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, (M, \rho), (M', \rho'))$ does as follows.

  1. Parse ω as (σ_P, θ).
  2. Verify whether θ is a non-repudiation information about the transaction with respect to the tuple $(\sigma_P, m_A,\ m_B, (M, \rho), (M', \rho'))$. If so, it continues; otherwise, it aborts and outputs $\perp$.
  3. Compute $(m_1, \cdots, m_k) \leftarrow H_0(m_A)$, and verify whether

  $$e(S_1, g) \cdot e(S_2^{-1}, \bar{u}' \prod_{j=1}^{k} \bar{u}_j^{m_j}) = e(\bar{g}_1, \bar{g}_2)$$

  holds. If so, it outputs $\top$; if not, it outputs $\perp$.

- $\mathbf{Tran_S}$: $\mathbf{Tran_S}(m, \mathsf{SK}_i, \mathsf{APK}, \mathsf{PM}, (M, \rho))$, where M is an $L \times N$ matrix and ρ a map from each row M_j of M to an attribute $\rho(j)$, does as follows:

  1. Compute $(m_1, \cdots, m_k) \leftarrow H_0(m)$,
  2. Choose a random exponent $r \in \mathbb{Z}_{n_1}$ and computes

  $$S_1 = \bar{g}_2^{x_1} \cdot (\bar{u}' \prod_{i=1}^{k} \bar{u}_i^{m_i})^r, \quad \text{and} \quad S_2 = g^r.$$

  3. Compute $(m'_1, \cdots, m'_k) \leftarrow H_0(m||S_1||S_2||\mathsf{PK}_i)$.
  4. Choose two random exponents $t_0, t_1 \in \mathbb{Z}_{n_1}$ and sets

  $$C'_0 = (\bar{g}_3/B)h^{t_0}, \pi_0 = ((\bar{g}_3/B)h^{t_0})^{t_0}, C'_1 = h^{t_1}, \pi_1 = ((\mathsf{APK}/B)^{-1}h^{t_1})^{t_1}.$$

  5. Choose $r' \leftarrow \mathbb{Z}_{n_1}$, and compute first $t = t_1 + t_2$ and then

  $$S'_1 = A^{x_3} \cdot (u' \prod_{j=1}^{k} u_j^{m'_j})^{r'} \cdot \hat{A}^t, \quad \text{and} \quad S'_2 = g^{r'}.$$

6. Choose a random vector $v \in \mathbb{Z}_{n_2}^N$, denoted $v = (s, v_2, \cdots, v_N)$. For each row M_j of M, it chooses a random $r_j \in \mathbb{Z}_{n_2}$.
7. Choose a random element $D \in G_T$, computes $sk = H_1(D)$, and uses sk as a symmetric key of SE to encrypt the bit strings $S_1||S_2||S_1'||S_2'||C_0'||C_1'||\pi_0||\pi_1$ and gains a ciphertext $\tilde{c}$.
8. The credential-protected package π is set as

$$\tilde{c},\ C_0 = De(g_1, g_1)^{\alpha s},\ C = g_1^s,\ C' = (g_1^{\gamma})^s,$$

$$C_j = (g_1^c)^{M_j \cdot v} h_{\rho(j)}^{-r_j},\ D_j = g_1^{r_j},\ \text{for } j = 1, \cdots, L.$$

- **Tran**$_{\mathsf{V}}$: **Tran**$_{\mathsf{V}}(m, \pi, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, \mathsf{CD}_S)$ does as follows.

1. Computes constants $\omega_j \in \mathbb{Z}_{n_2}$ such that $\sum_{\rho(j) \in S} \omega_j M_j = (1, 0, \cdots, 0)$. It then computes

$$\hat{e}(C, K)\hat{e}(C', K')^{-1} / \prod_{\rho(j) \in S} (\hat{e}(C_j, K'')\hat{e}(D_j, K_{\rho(j)}))^{\omega_j} = \hat{e}(g_1, g_1)^{\alpha s}.$$

2. Recover D as $C_0 / \hat{e}(g_1, g_2)^{\alpha s}$, compute $sk = H_1(D)$ and uses sk as a symmetric key of SE to decrypt the ciphertext c to gain a full signature σ.

- **Ver**: **Ver**$(m, \sigma, \mathsf{PK}_i, \mathsf{APK})$ does as follows.

1. Parse σ as $(S_1, S_2, S_1', S_2', C_0, C_1, \pi_0, \pi_1)$, compute $(m_1, \cdots, m_k) \leftarrow H_0(m)$, and verify whether

$$e(S_1, g) \cdot e(S_2^{-1}, \bar{u}' \prod_{j=1}^{k} \bar{u}_j^{m_j}) = e(\bar{g}_1, \bar{g}_2)$$

holds. If so, output $\top$; if not, output $\perp$.
2. Check whether

$$e(C_0', C_0'/(\bar{g}_3/B)) = e(h, \pi_0) \ \text{ and } \ e(C_1', C_1'/(\mathsf{APK}/B)) = e(h, \pi_1)$$

hold. If either does not hold, output $\perp$.

3. Compute $(m'_1, \cdots, m'_k) \leftarrow H_0(m||S_1||S_2||\mathsf{PK}_i)$ and $C'_2 = C'_0 \cdot C'_1$, and verify whether

$$e(A, BC'_2) = e(S'_1, g) \cdot e((S'_2)^{-1}, u' \prod_{j=1}^{k} u_j^{m'_j})$$

holds. If so, output $\top$; if not, output $\perp$.

- $\mathbf{Res}_{\mathsf{User}}$: $\mathbf{Res}_{\mathsf{User}}(m_A, m_B, \omega, \mathsf{SK}_j, \mathsf{PK}_i, \mathsf{APK}, \mathsf{PM}, (M, \rho), (M', \rho'))$ does as follows.

1. Parses PK_j as $(\tilde{g}_1, \tilde{g}_2, \tilde{g}_3, \tilde{u}', \tilde{u}_1, \cdots, \tilde{u}_k)$ and $\mathsf{SK}_j := (\tilde{g}_2^{x'_1}, A^{x_3})$.
2. Compute $(m_1, \cdots, m_k) \leftarrow H_0(m_B)$,
3. Choose a random exponent $r \in \mathbb{Z}_{n_1}$ and computes

$$S_1 = \tilde{g}_2^{x'_1} \cdot (\bar{u}' \prod_{i=1}^{k} \tilde{u}_i^{m_i})^r, \quad \text{and} \quad S_2 = g^r.$$

4. Compute $(m'_1, \cdots, m'_k) \leftarrow H_0(m||S_1||S_2||\mathsf{PK}_j)$.
5. Choose two random exponents $t_0, t_1 \in \mathbb{Z}_{n_1}$ and sets

$$C'_0 = (\tilde{g}_3/B)h^{t_0}, \pi_0 = ((\tilde{g}_3/B)h^{t_0})^{t_0}, C'_1 = h^{t_1}, \pi_1 = ((\mathsf{APK}/B)^{-1}h^{t_1})^{t_1}.$$

6. Choose $r' \leftarrow \mathbb{Z}_{n_1}$, and compute first $t = t_1 + t_2$ and then

$$S'_1 = A^{x_3} \cdot (u' \prod_{j=1}^{k} u_j^{m'_j})^{r'} \cdot \hat{A}^t, \quad \text{and} \quad S'_2 = g^{r'}.$$

7. Set $\sigma_B := (S_1, S_2, S'_1, S'_2, C_0, C_1, \pi_0, \pi_1)$.
8. Set $\mathsf{request} := (\omega, m_B, \sigma_B, \mathsf{PK}_j, (M, \rho))$.

- $\mathbf{Res}_{\mathsf{TTP}}$: $\mathbf{Res}_{\mathsf{TTP}}(m_A, \mathsf{PK}_i, (M', \rho'), \mathsf{request}, \mathsf{ASK})$ does as follows.

1. Parse $\mathsf{request}$ as $(\omega, m_B, \sigma_B, \mathsf{PK}_j, (M, \rho))$ and ω as σ_P and θ. If the non-repudiation information θ is not with respect to the tuple $(\sigma_P, m_A, m_B, (M, \rho), (M', \rho'))$, outputs $\perp$.

2. Parse σ_P as (S_1, S_2), compute $(m_1, \cdots, m_k) \leftarrow H_0(m_A)$, and verify whether

$$e(S_1, g) \cdot e(S_2^{-1}, \bar{u}' \prod_{j=1}^{k} \bar{u}_j^{m_j}) = e(\bar{g}_1, \bar{g}_2)$$

holds. If not, output $\perp$.
3. Check whether $\mathbf{Ver}(m_B, \sigma_B, \mathsf{PK}_j, \mathsf{APK}) = \top$. If not, output $\perp$.
4. Compute $(m'_1, \cdots, m'_k) \leftarrow H_0(m_A||S_1||S_2||\mathsf{PK}_i)$.
5. Choose two random exponents $t_0, t_1 \in \mathbb{Z}_{n_1}$ and sets

$$C'_0 = h^{t_0}, \pi_0 = ((\bar{g}_3/B)^{-1} h^{t_0})^{t_0}, C'_1 = (\mathsf{APK}/B) h^{t_1}, \pi_1 = ((\mathsf{APK}/B) h^{t_1})^{t_1}.$$

6. Choose $r' \leftarrow \mathbb{Z}_{n_1}$, and compute first $t = t_1 + t_2$ and then

$$S'_1 = A^y \cdot (u' \prod_{j=1}^{k} u_j^{m'_j})^{r'} \cdot \hat{A}^t, \quad \text{and} \quad S'_2 = g^{r'}.$$

7. Choose a random vector $v \in \mathbb{Z}_{n_2}^{N'}$, denoted $v = (s, v_2, \cdots, v_{N'})$. For each row M_j of M, it chooses a random $r_j \in \mathbb{Z}_{n_2}$.
8. Choose a random element $D \in G_T$, computes $sk = H_1(D)$, and uses sk as a symmetric key of SE to encrypt the bit strings $S_1||S_2||S'_1||S'_2||C'_0||C'_1||\pi_0||\pi_1$ and gains a ciphertext $\tilde{c}_A$.
9. The credential-protected package π_A is set as

$$\tilde{c}_A, \ C_0 = De(g_1, g_1)^{\alpha s}, \ C = g_1^s, \ C' = (g_1^{\gamma})^s,$$

$$C_j = (g_1^c)^{M'_j \cdot v} h_{\rho'(j)}^{-r_j}, \ D_j = g_1^{r_j}, \ \text{for } j = 1, \cdots, L'.$$

10. Choose a random vector $v' \in \mathbb{Z}_{n_2}^{N}$, denoted $v' = (s', v'_2, \cdots, v'_N)$. For each row M'_j of M', it chooses a random $r'_j \in \mathbb{Z}_{n_2}$.
11. Choose a random element $D' \in G_T$, computes $sk' = H_1(D')$, and uses sk' as a symmetric key of SE to encrypt the σ_B and gains a ciphertext $\tilde{c}_B$.
12. The credential-protected package π_B is set as

$$\tilde{c}_B, \ C_0 = De(g_1, g_1)^{\alpha s'}, \ C = g_1^{s'}, \ C' = (g_1^{\gamma})^{s'},$$

$$C_j = (g_1^c)^{M_j \cdot v'} h_{\rho(j)}^{-r'_j}, \ D_j = g_1^{r'_j}, \ \text{for } j = 1, \cdots, L.$$

13. The output is (π_A, π_B).

Algorithms:	$\mathbf{Tran_P}$	$\mathbf{TPVer}$	$\mathbf{Tran_S}$	$\mathbf{Tran_V}$	$\mathbf{Ver}$
Cost:	2E	3P	1SE, $(12+3L)$E	1SD, $(2s+2)$P, (s)E	10P

Table 1: Costs of algorithms in our instantiation of ABOFE.

5.1. Efficency Analysis

Since pairing and exponentiation operations take more time than multiplication operations do, we will simply ignore the costs of multiplication computations here. Let $\mathrm{SE}, \mathrm{SD}, \mathrm{P}$ and E denote a symmetric encryption operation, a symmetric decryption operation and a pairing operation, and an exponentiation operation, respectively. Let M be an $L \times N$ matrix, ρ a map from each row M_j of M to an attribute $\rho(j)$, and S a set of attributes that satisfies the access structure (M, ρ). Denote s as the number of elements of the set $\{1 \leq j \leq L | \rho(j) \in S\}$.

The costs of our instantiation of ABOFE is presented in Table 1, in which $\mathbf{Tran_S}$ is evaluated with respect to the access structure (M, ρ), and $\mathbf{Tran_V}$ is evaluated with respect to the set of attributes S.

6. Conclusion

We proposed the notion of attribute-based optimistic fair exchange, and gave a formal security model. We then proposed a generic construction of PAOFE, and proved its security under the proposed model in the standard model. Since ABOFE can be used to solve the fair exchange problem when the attributes of the participants need to be taken into account, ABOFE can be viewed as an extension of OFE in the attribute-based setting.

Acknowledgement

We would like to thank the anonymous reviewers who have helped us to improve the clarity and quality of this paper.

References

[1] N. Asokan, M. Schunter, M. Waidner, Optimistic protocols for fair exchange, in: Proc. 4th ACM Conference on Computer and Communications Security, 1997, pp. 8–17.

[2] J. A. Garay, M. Jakobsson, P. D. MacKenzie, Abuse-free optimistic contract signing, in: Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology, CRYPTO '99, Springer-Verlag, London, UK, UK, 1999, pp. 449–466. URL: http://dl.acm.org/citation.cfm?id=646764.703965.

[3] O. Markowitch, S. Kremer, An optimistic non-repudiation protocol with transparent trusted third party, in: in: Information Security Conference 2001, Lecture Notes in Computer Science, Springer-Verlag, 2001, pp. 363–378.

[4] X. Huang, Y. Mu, W. Susilo, W. Wu, J. Zhou, R. H. Deng, Preserving transparency and accountability in optimistic fair exchange of digital signatures, IEEE Transactions on Information Forensics and Security 6 (2011) 498–512.

[5] H. Zhu, F. Bao, Stand-alone and setup-free verifiably committed signatures, in: CT-RSA, volume 3860 of *Lecture Notes in Computer Science*, 2006, pp. 159–173.

[6] B. B. Anderson, J. V. Hansen, P. B. Lowry, S. L. Summers, Standards and verification for fair-exchange and atomicity in e-commerce transactions, Information Sciences 176 (2006) 1045–1066.

[7] Q. Huang, D. S. Wong, W. Susilo, Group-oriented fair exchange of signatures, Information Sciences 181 (2011) 3267–3283.

[8] Q. Huang, D. S. Wong, W. Susilo, The construction of ambiguous optimistic fair exchange from designated confirmer signature without random oracles, Information Sciences 228 (2013) 222–238.

[9] N. Asokan, V. Shoup, M. Waidner, Optimistic fair exchange of digital signatures, Advances in Cryptology-Eurocrypt 1998, Lecture Notes in Computer Science 1403 (1998) 591 – 606.

[10] J. Camenisch, I. Damgård, Verifiable encryption, group encryption, and their applications to separable group signatures and signature sharing schemes, in: Proceedings of the 6th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology, ASI-

ACRYPT '00, Springer-Verlag, London, UK, 2000, pp. 331–345. URL: http://dl.acm.org/citation.cfm?id=647096.716992.

[11] D. Boneh, C. Gentry, B. Lynn, H. Shacham, Aggregate and verifiably encrypted signatures from bilinear maps, Advances in cryptology - Eurocrypt 2003, Lecture Notes in Computer Science 2656 (2003) 416–432.

[12] S. Lu, R. Ostrovsky, A. Sahai, H. Shacham, B. Waters, Sequential aggregate signatures and multisignatures without random oracles, in: EUROCRYPT, 2006., Springer, 2006, pp. 465–485.

[13] J. Zhang, J. Mao, A novel verifiably encrypted signature scheme without random oracle, in: Proceedings of the 3rd international conference on Information security practice and experience, ISPEC'07, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 65–78. URL: http://dl.acm.org/citation.cfm?id=1759508.1759518.

[14] Z. Shao, Certificate-based verifiably encrypted signatures from pairings, Information Sciences 178 (2008) 2360–2373.

[15] M. Rückert, D. Schröder, Security of verifiably encrypted signatures and a construction without random oracles, in: Proceedings of the 3rd International Conference Palo Alto on Pairing-Based Cryptography, Pairing '09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 17–34.

[16] Y. Dodis, L. Reyzin, Breaking and repairing optimistic fair exchange from podc 2003, in: Proceedings of the 3rd ACM workshop on Digital rights management, DRM '03, ACM, New York, NY, USA, 2003, pp. 47–54. URL: http://doi.acm.org/10.1145/947380.947387. doi:http://doi.acm.org/10.1145/947380.947387.

[17] Y. Dodis, P. J. Lee, D. H. Yum, Optimistic fair exchange in a multi-user setting, in: Proceedings of the 10th international conference on Practice and theory in public-key cryptography, PKC'07, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 118–133. URL: http://dl.acm.org/citation.cfm?id=1760564.1760577.

[18] Q. Huang, G. Yang, D. S. Wong, W. Susilo, Efficient optimistic fair exchange secure in the multi-user setting and chosen-key model

without random oracles, in: Proceedings of the 2008 The Cryptopgraphers' Track at the RSA conference on Topics in cryptology, CT-RSA'08, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 106–120. URL: http://dl.acm.org/citation.cfm?id=1791688.1791698.

[19] B. Barak, R. Canetti, J. B. Nielsen, R. Pass, Universally Composable Protocols with Relaxed Set-Up Assumptions, in: FOCS, IEEE Computer Society, 2004, pp. 186–195.

[20] A. Sahai, B. Waters, Fuzzy identity-based encryption, in: R. Cramer (Ed.), EUROCRYPT, volume 3494 of *Lecture Notes in Computer Science*, Springer, 2005, pp. 457–473.

[21] B. Waters, Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization, in: D. Catalano, N. Fazio, R. Gennaro, A. Nicolosi (Eds.), Public Key Cryptography, volume 6571 of *Lecture Notes in Computer Science*, Springer, 2011, pp. 53–70.

[22] A. Beimel, Secure Schemes for Secret Sharing and Key Distribution, Ph.D. thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.

[23] A. B. Lewko, B. Waters, New proof methods for attribute-based encryption: Achieving full security through selective techniques, in: R. Safavi-Naini, R. Canetti (Eds.), CRYPTO, volume 7417 of *Lecture Notes in Computer Science*, Springer, 2012, pp. 180–198.

[24] Q. Huang, G. Yang, D. S. Wong, W. Susilo, Ambiguous optimistic fair exchange, in: Advances in Cryptology - ASIACRYPT 2008, 14th International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, Australia, December 7-11, 2008. Proceedings, 2008, pp. 74–89.

[25] B. Waters, Efficient identity-based encryption without random oracles, in: R. Cramer (Ed.), EUROCRYPT, volume 3494 of *Lecture Notes in Computer Science*, Springer, 2005, pp. 114–127.

[26] H. Shacham, B. Waters, Efficient ring signatures without random oracles, in: T. Okamoto, X. Wang (Eds.), Public Key Cryptography, volume 4450 of *Lecture Notes in Computer Science*, Springer, 2007, pp. 166–180.