2012

Perfect ambiguous optimistic fair exchange

Yang Wang
University of Wollongong, ywang@uow.edu.au

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

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

Publication Details

Wang, Y., Au, M. & Susilo, W. (2012). Perfect ambiguous optimistic fair exchange. Lecture Notes in Computer Science, 7618 (2012), 142-153.

Perfect ambiguous optimistic fair exchange

Abstract
Protocol for fair exchange of digital signatures is essential in many applications including contract signing, electronic commerce, or even peer-to-peer file sharing. In such a protocol, two parties, Alice and Bob, would like to exchange digital signatures on some messages in a fair way. It is known that a trusted arbitrator is necessary in the realization of such a protocol. We identify that in some scenarios, it is required that prior to the completion of the protocol, no observer should be able to tell whether Alice and Bob are conducting such an exchange. Consider the following scenario in which Apple engages Intel in an exchange protocol to sign a contract that terminates their OEM agreement. The information would be of value to a third party (such as the stock broker, or other OEM companies). If the protocol transcript can serve as an evidence that such a communication is in progress, any observer of this communication, including the employees of both companies, would be tempted to capture the transcript and sell it to outsiders. We introduce a new notion called perfect ambiguous optimistic fair exchange (PAOFE), which is particularly suitable to the above scenario. PAOFE fulfils all traditional requirements of cryptographic fair exchange of digital signatures and, in addition, guarantees that the communication transcript cannot be used as a proof to convince others that the protocol is in progress. Specifically, we formalize the notion of PAOFE and present a rigorous security model in the multi-user setting under the chosen-key attack. We also present a generic construction of PAOFE from existing cryptographic primitives and prove that our proposal is secure with respect to our definition in the standard model. 2012 Springer-Verlag.

Keywords
exchange, fair, perfect, optimistic, ambiguous

Disciplines
Physical Sciences and Mathematics

Publication Details
Wang, Y., Au, M. & Susilo, W. (2012). Perfect ambiguous optimistic fair exchange. Lecture Notes in Computer Science, 7618 (2012), 142-153.

Perfect Ambiguous Optimistic Fair Exchange

Yang Wang, Man Ho Au, and Willy Susilo*

Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Australia
{yw990}@uowmail.edu.au {aau, wsusilo}@uow.edu.au

Abstract. Protocol for fair exchange of digital signatures is essential in many applications including contract signing, electronic commerce, or even peer-to-peer file sharing. In such a protocol, two parties, Alice and Bob, would like to exchange digital signatures on some messages in a fair way. It is known that a trusted arbitrator is necessary in the realization of such a protocol.
We identify that in some scenarios, it is required that prior to the completion of the protocol, no observer should be able to tell whether Alice and Bob are conducting such an exchange. Consider the following scenario in which Apple engages Intel in an exchange protocol to sign a contract that terminates their OEM agreement. The information would be of value to a third party (such as the stock broker, or other OEM companies). If the protocol transcript can serve as an evidence that such a communication is in progress, any observer of this communication, including the employees of both companies, would be tempted to capture the transcript and sell it to outsiders.
We introduce a new notion called *perfect ambiguous optimistic fair exchange* (PAOFE), which is particularly suitable to the above scenario. PAOFE fulfils all traditional requirements of cryptographic fair exchange of digital signatures and, in addition, guarantees that the communication transcript cannot be used as a proof to convince others that the protocol is in progress. Specifically, we formalize the notion of PAOFE and present a rigorous security model in the multi-user setting under the chosen-key attack. We also present a generic construction of PAOFE from existing cryptographic primitives and prove that our proposal is secure with respect to our definition in the standard model.

1 Introduction

Consider a scenario in which Apple engages Intel in a fair exchange protocol to sign a contract that pays an amount of money for the early termination of the use of Intel technology in the next generation of Macbook and iMac desktop computers. In this situation, reveal of the contract, or leakage of the information about this contract, prior to its effective date will be potentially harmful to the companies. For instance, Apple may be reluctant to expose prematurely the changes it is introducing to its next generation products, which may possibly affect the sales of the current generation of the products. On the other hand, the potential termination of cooperation with Apple may lead to a decline of Intel's shares value. Therefore, it is necessary that the fair exchange protocol should not leak any information about the signatures being exchanged.

To the best of our knowledge, ambiguous optimistic fair exchange (AOFE) [14] is the closest cryptographic solution to the above problem. An AOFE protocol comprises three parties, namely, signer Alice, verifier Bob, and a semi-trusted third party known as the "arbitrator". In a typically execution of an AOFE protocol, Alice delivers a "commitment" of her signature, called ambiguous partial signature, to Bob. Upon successful verification of the ambiguous partial signature, Bob delivers his full signature to Alice. After verifying the full signature from Bob, Alice sends to Bob her own full signature. This completes the protocol.

Bob can approach the arbitrator for assistance in the situation in which Alice refuses to send her full signature at the end of the exchange protocol. The ambiguous partial signature is designed

* This work is supported by ARC Future Fellowship FT0991397.

in such a way that the arbitrator can turn it into Alice's full signature, which is indistinguishable to a "real" signature created by Alice. In this way, as long as the arbitrator is trusted to carry out its duty, Bob can always be assured he can obtain a full signature from Alice, either from Alice or the arbitrator. In addition, the arbitrator is not required to take part in typical executions of the protocol.

AOFE differs from traditional optimistic fair exchange (OFE) [1] in the sense that the ambiguous partial signature does not reveal the identity of its creator. Specifically, in OFE, everyone can verify that Alice has created a commitment of her signature in the first step. This may create an unfair situation to Alice as Bob can simply use Alice's commitment as a mean to his advantage. For instance, if Alice's signature represents her contract tender for Bob's service, Bob can use Alice's commitment as a way to ask for a higher price from another party. On the other hand, the ambiguous partial signature in AFOE has the extra property that it can be created by either Alice or Bob. Thus, while Bob can be assured that this is Alice's commitment of her signature, he cannot convince anybody that this is Alice's commitment since he could have been the creator of the ambiguous partial signature as well. Nonetheless, in AOFE, the arbitrator knows who is the creator of the ambiguous signature.

Unfortunately, AOFE is inadequate to the aforementioned problem we raised earlier. If AOFE is employed in the above scenario, Apple will transmit the ambiguous partial signature to Intel on the contract of the termination of the use of Intel technology in its next generation of computers as the first step of the exchange. This ambiguous partial signature itself leaks sufficient information to be valuable. The reason is that in this scenario, it does not matter who is the signer of this contract. The valuable information to an outsider is that these two companies are discussing about a potential termination, which is the partial signature. The ambiguous partial signature created by Apple or Intel is sufficient evidence to prove the authenticity of the information. At the first sight, one may think that providing a secure channel between the parties would be sufficient in the above scenario. Nevertheless, this approach has a huge drawback. To build a secure channel between any two parties is known to be extremely expensive, and therefore, this approach will not be feasible in practice.

One key observation about the existing exchange protocol is that the ambiguous partial signature in AOFE, as well as the regular partial signature in OFE, is indeed publicly verifiable. This is not strictly a necessary functional requirement of an exchange protocol. In fact, this may have an undesirable effect as illustrated in our case earlier. In general, if Bob is known to be trustworthy, for example, if Bob is a government department, then malicious observer Oven who obtains an ambiguous partial signature submitted to Bob knows the intention of Alice. Besides, we make the observation that the arbitrator in AOFE knows who the creator of an ambiguous partial signature is, and is capable of converting it into a full signature. A high level of trust has to be placed on the arbitrator.

Hence, we introduce a new notion, called *Perfect Ambiguous Optimistic Fair Exchange* (PAOFE), as a practical cryptographic solution to the aforementioned scenario. Indeed, our solution builds on top of AOFE and it also fulfills all the security requirements of an AOFE. In addition, PAOFE enjoys a new property called *Perfect Ambiguity* in which the equivalent of an "ambiguous partial signature" leaks no information about the actual signer, intended recipient and the signature itself, and not even in the view of the arbitrator. Thus, no outsider can tell if an exchange is in progress.

1.1 Related Work

Optimistic fair exchange (OFE), well-known for solving the fair exchange problem, was first introduced by Asokan, Schunter and Waidner [1]. Since then, extensive research on the issue [5,6,9–13, 16–21] have been conducted. An optimistic fair exchange protocol consists of a signer, a verifier, and a semi-trusted third party named "arbitrator". Typically such a protocol can be conducted in three message flows as follows. Firstly, Alice the signer initiates the protocol by delivering a partial signature to Bob the verifier. A valid partial signature not only serves as an evidence to Bob that Alice has committed to endorse a certain message, but also assures Bob that he will receive Alice's full signature at the end of the protocol. This is due to the property that the arbitrator has the

power to convert a valid partial signature into a full signature. In the second step, Bob delivers his full signature to Alice. Later, if Alice is honest, she will send her full signature to Bob in the third step, and the exchange process finishes. Note that in normal circumstance, no participation from the arbitrator is required and thus, the term 'optimistic'. On the other hand, the arbitrator is trusted in two senses. Alice trusts the arbitrator would not convert her partial signature into a full signature unless Bob submits his full signature. At the same time, Bob trusts the arbitrator that if he submits his full signature, the arbitrator would convert Alice's partial signature into a full one.

Security of early optimistic fair exchange protocols are studied in the single-user setting, i.e., there is only one signer, one verifier along with one arbitrator. The first formal security model was proposed in [1, 2], which considered the cases when the signer or verifier cheats but ignored the one that an arbitrator itself might be potentially dishonest. It was further extended by Dodis and Reyzin [9] to a more generalized model in which a possibly cheating arbitrator is discussed. Since there are many users in the real world, it would be practical to allow a number of signers sharing the same arbitrator. In 2007, Dodis, Lee and Yum [8] considered optimistic fair exchange in the multi-user setting, where there are many signers and verifiers along with one arbitrator in a system. In the multi-user setting, dishonest users are allowed to collude to cheat another user. Dodis et al. [8] pointed out that the security of an OFE in the single-user setting does not necessarily guarantee that in the multi-user setting. Furthermore, they proposed a formal definition of OFE in the multi-user setting and a generic construction that is secure in this more practical model was also proposed [8].

In an orthogonal dimension, most optimistic fair exchange protocols are studied in the *certified-key* model (also known as the *registered-key* model [3]). In this model, to use a public key, the adversary must show its knowledge of the corresponding private key, and is only allowed to make queries with respect to the registered public keys. In 2008, Huang, Yang, Wong and Susilo [15] studied the security of optimistic fair exchange in the multi-user setting and *chosen-key model*, where the adversary can choose its public key arbitrarily probably without knowing the corresponding private keys. The adversary is allowed to make queries with respect to these arbitrarily chosen public keys. They demonstrated, through an example, that a provably secure fair exchange in the certified-key model may not be secure in the chosen-key model. Furthermore, a generic optimistic fair exchange construction secure in this model was proposed.

In 2008, Huang, Yang, Wong and Susilo [14] introduced a new security notion called the *signer ambiguity* to OFE, which requires that Bob is able to generate a partial signature that is indistinguishable to a real partial signature generated by Alice. With this property, Bob will not be able to convince any outsiders that a partial signature was indeed generated by Alice. They named OFE with this new security property *Ambiguous Optimistic Fair Exchange*. Besides, they proposed a formal security model for AOFE in the multi-user setting and chosen key model, a generic construction of AOFE and the first efficient AOFE scheme.

1.2 Our Contributions

In this paper we make the following contributions.

1. We propose the notion of *Perfect Ambiguous Optimistic Fair Exchange*, which allows a signer Alice to generate a partial signature in such a way that no outsider, not even the arbitrator, is able to infer any useful information about the signature. Indeed, a partial signature in PAOFE generated by the signer Alice with Bob being the receiver is indistinguishable to a random bit string chosen from the signature space. In other words, any partial signature is indistinguishable to a partial signature on a random message with respect to a random signer and receiver. To realize this notion, Bob's secret key is required in the verification of the partial signature in PAOFE. Thus, only Bob is able to verify the partial signature, and an outsider gains nothing about the transaction. Both the identities of the signer and receiver and the content of an transaction are perfectly hidden.

2. We define a security model for PAOFE in the multi-user setting under chosen-key attack. Our model captures the existing security requirements for AOFE, namely, signer ambiguity, resolution ambiguity, security against signers, security against verifiers and security against the arbitrator. In addition, PAOFE covers an additional requirement: *perfect ambiguity*. It is required that any user can generate a partial signature whose distribution is indistinguishable from that of a partial signatures generated by Alice. In other words, a specific partial signature generated by Alice with recipient Bob is indistinguishable from a partial signature uniformly randomly chosen from the whole signature space.
3. We propose a generic construction of PAOFE from two well established cryptographic primitives, namely, AOFE and key-private encryption 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.

1.3 Paper Organization

In the next section, we review the notions and security models of public key encryption and AOFE respectively. In Section 3, a formal definition of PAOFE, together with the security model in the multi-user and chosen key setting is proposed. Then, we propose a generic construction of PAOFE and also provide the security proof of our scheme under our model in Section 4. Finally, we conclude the paper in Section 5.

2 Building Blocks

Throughout the paper, the following notations are used. For a finite set $\mathcal{S}$, $s \leftarrow \mathcal{S}$ denotes that a 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. We use $[A_1(\mathsf{in}_1) \rightarrow \mathsf{out}_1] \overset{\mathbf{P}}{\Longleftrightarrow} [A_2(\mathsf{in}_2) \rightarrow \mathsf{out}_2]$ to denote that two PPT algorithms A_1 and A_2 outputs out_1 and out_2 respectively upon the completion of the protocol $\mathbf{P}$ in which A_1 takes as input in_1 and A_2 takes as input in_2.

2.1 Encryption

A public key encryption scheme $\mathcal{E}$ consists of three algorithms: $\mathcal{E} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$. We consider *indistinguishability of encryptions against adaptive chosen ciphertext attacks*, denoted by IE-CCA [4]. It is identical to the more widely used notion IND-CCA [7]. Here we just adopt the notion IE-CCA, as the authors did in [4], to distinguish with another kind of indistinguishability to be considered next. For an efficient algorithm $\mathcal{A}$, which runs in two stages of find and guess, we define the adversary's advantage $\mathsf{IE\text{-}Adv}^{\mathcal{E}}_{\mathcal{A}}(k)$ as

$$\left| \Pr \left[b = \tilde{b} \;\middle|\; \begin{array}{l} (ek, dk) \leftarrow \mathsf{Kg}(1^\kappa),\ (m_0, m_1, \alpha) \leftarrow \mathcal{A}^{O_{\mathsf{Dec}}}(ek, \mathsf{find}), \\ b \leftarrow \{0,1\},\ c_b \leftarrow \mathsf{Enc}_{ek}(m_b),\ \tilde{b} \leftarrow \mathcal{A}^{O_{\mathsf{Dec}}}(c_b, \alpha, \mathsf{guess}) \end{array} \right] - \frac{1}{2} \right|$$

where $\mathcal{A}$ is allowed to invoke the decryption oracle $O_{\mathsf{Dec}}(\cdot)$ at any point with the only restriction of not querying c_b during the guess stage. The encryption scheme $\mathcal{E}$ is said to be IE-CCA secure if the function $\mathsf{IE\text{-}Adv}^{\mathcal{E}}_{\mathcal{A}}(k)$ is negligible for any probabilistic polynomial-time time adversary $\mathcal{A}$.

It is well-known that the above security notion of encryption schemes captures the strongest message-privacy property. Given a challenge ciphertext, no information about the underlying message (plaintext) will be leaked in an IE-CCA secure encryption scheme. However, it does not take into account the privacy of keys, that is, given a ciphertext, the above notion does not include the privacy of the public key of the recipient. To hide the information about the public key under which an encryption is conducted, we consider *indistinguishability of keys under adaptive chosen ciphertext attacks*, denoted by IK-CCA [4]. For an efficient algorithm $\mathcal{A}$, which runs in two stages of find and guess, we define the adversary's advantage $\mathsf{IK\text{-}Adv}^{\mathcal{E}}_{\mathcal{A}}(k)$ as

$$\left|\Pr\left[b=\tilde{b}\;\middle|\;\begin{array}{l}(ek_0,dk_0)\leftarrow \mathsf{Kg}(1^\kappa),\ (ek_1,dk_1)\leftarrow \mathsf{Kg}(1^\kappa),\\ (m,\alpha)\leftarrow \mathcal{A}^{\mathcal{D}_{dk_0}(\cdot),\mathcal{D}_{dk_1}(\cdot)}(ek_0,ek_1,\mathsf{find}),\ b\leftarrow\{0,1\},\\ c_b\leftarrow \mathsf{Enc}_{ek_b}(m),\ \tilde{b}\leftarrow \mathcal{A}^{\mathcal{D}_{dk_0}(\cdot),\mathcal{D}_{dk_1}(\cdot)}(c_b,\alpha,\mathsf{guess})\end{array}\right]-\frac{1}{2}\right|$$

where $\mathcal{A}$ is allowed to invoke the decryption oracles $\mathcal{D}_{dk_0}(\cdot)$ and $\mathcal{D}_{dk_1}(\cdot)$ at any point with the only restriction of not querying c_b to neither oracle during the guess stage. The scheme $\mathcal{E}$ is said to be IK-CCA secure if the function $\mathsf{IK\text{-}Adv}^{\mathcal{E}}_{\mathcal{A}}(k)$ is negligible for any probabilistic polynomial-time adversary $\mathcal{A}$.

Though the goals of message-privacy and key-privacy are orthogonal, it is very desirable, from a practical point of view, that an encryption scheme satisfies both sides. To guarantee both the message-privacy and key-privacy properties at the same time, we combine the above two security notions into one.

Definition 1. *An encryption scheme $\mathcal{E}$ consisting of three algorithms $\mathcal{E}=(\mathsf{Kg},\mathsf{Enc},\mathsf{Dec})$ is said to be* IE-IK-CCA *secure if for any probabilistic polynomial-time algorithm $\mathcal{A}$, the advantage of $\mathcal{A}$ $\mathsf{Adv}^{\mathsf{IE\text{-}IK}}_{\mathcal{A}}(\kappa)$ is negligible in κ, where $\mathsf{Adv}^{\mathsf{IE\text{-}IK}}_{\mathcal{A}}(\kappa)$ is defined as*

$$\left|\Pr\left[b=\tilde{b}\;\middle|\;\begin{array}{l}(ek,dk)\leftarrow \mathsf{Kg}(1^\kappa),(m,\alpha)\leftarrow \mathcal{A}^{O_{\mathsf{Dec}}}(ek,\mathsf{find}),\ b\leftarrow\{0,1\},\\ c_b\leftarrow\left\{\begin{array}{ll}\mathsf{Enc}_{ek}(m) & if\ b=0\\ c'\leftarrow\mathcal{C} & if\ b=1\end{array}\right.,\ \tilde{b}\leftarrow \mathcal{A}^{O_{\mathsf{Dec}}}(c_b,\alpha,\mathsf{guess})\end{array}\right]-\frac{1}{2}\right|$$

where $\mathcal{C}$ is the whole ciphertext space with respect to any message and any public key, and $\mathcal{A}$ is allowed invoke the decryption oracle $O_{\mathsf{Dec}}(\cdot)$ at any point with the only restriction of not querying c_b during the guess *stage.*

It is easy to see that any public key encryption scheme that is both IE-CCA secure and IK-CCA secure will be IE-IK-CCA secure. Since Cramer-Shoup encryption scheme [7] is both IE-CCA secure and IK-CCA secure [4], it is IE-IK-CCA secure.

2.2 Ambiguous Optimistic Fair Exchange

We review the notion and security model of the ambiguous optimistic fair exchange protocol introduced in [14].

Definition 2. *An ambiguous optimistic fair exchange scheme involves the users (signers and verifiers) and the arbitrator, and consists of the following (probabilistic) polynomial-time algorithms:*

- PMGen*: On input 1^κ where κ is a security parameter, it outputs a system parameter PM.*
- $\mathsf{Setup}^{\mathsf{TTP}}$*: On input PM, the algorithm generates a secret key ASK, and a public key APK of the arbitrator.*
- $\mathsf{Setup}^{\mathsf{User}}$*: On input PM 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*: $\mathsf{Sig}(M,SK_i,PK_i,PK_j,APK)$, outputs a (full) signature σ on M of user U_i with the designated verifier U_j, where message M is chosen by user U_i from the message space $\mathcal{M}$ defined under PK_i, while $\mathsf{Ver}(M,\sigma,PK_i,PK_j,APK)$ outputs $\top$ or $\bot$, indicating σ is U_i's valid full signature on M with the designated verifier U_j or not.*
- PSig *and* PVer*: These are partial signing and verification algorithms respectively. $\mathsf{PSig}(M,SK_i,PK_i,PK_j,APK)$ outputs a partial signature σ_P, while $\mathsf{PVer}(M,\sigma_P,\mathbf{PK},APK)$ outputs $\top$ or $\bot$, where $\mathbf{PK}=\{PK_i,PK_j\}$.*
- Res*: This is the resolution algorithm. $\mathsf{Res}(M,\sigma_P,ASK,\mathbf{PK})$, where $\mathbf{PK}=\{PK_i,PK_j\}$, outputs a full signature σ, or $\bot$ indicating the failure of resolving a partial signature.*

Resolution ambiguity property states that

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

The security of an ambiguous optimistic fair exchange scheme consists of four aspects: signer ambiguity, 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.

SIGNER AMBIGUITY. We require that any probabilistic polynomial-time distinguisher D succeeds with at most negligible probability greater than 1/2 in the following experiment.

$$\begin{aligned} PM &\leftarrow \mathsf{PMGen}(1^k) \\ (ASK, APK) &\leftarrow \mathsf{Setup}^{\mathsf{TTP}}(PM) \\ (M, (SK_0, PK_0), (SK_1, PK_1), \delta) &\leftarrow D^{O_{\mathsf{Res}}}(APK) \\ b &\leftarrow \{0,1\} \\ \sigma_P &\leftarrow \mathsf{PSig}(M, SK_b, PK_b, PK_{1-b}, APK) \\ b' &\leftarrow D^{O_{\mathsf{Res}}}(\sigma_P, \delta) \\ \text{success of } A &:= [b' = b \wedge (M, \sigma_P, \{PK_0, PK_1\}) \notin Query(D, O_{\mathsf{Res}})] \end{aligned}$$

where δ is D's state information, oracle O_{Res} takes as input a valid partial signature σ_P of user U_i on message M with respect to verifier U_j (i.e. (M, σ_P, PK_i, PK_j) such that $\mathsf{PVer}(M, \sigma_P, \{PK_i, PK_j\}, APK) = \top$), and outputs a full signature σ on M under PK_i, PK_j, and $Query(D, O_{\mathsf{Res}})$ is the set of valid queries D issued to the resolution oracle. That is to say, given a partial signature σ_P from a signer A, a verifier B should not be able to convince others that A was indeed the signer of σ_P, as B can generate partial signatures that look indistinguishable from those generated by A.

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

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

where oracle O_{Res} is described in the previous experiment, oracle O^B_{PSig} takes as input (M, PK_i) and outputs a signature on M with respect to PK_i and PK_B generated using SK_B, and $Query(\mathcal{A}, O^B_{\mathsf{PSig}})$ is the set of queries made by $\mathcal{A}$ to oracle O^B_{PSig}. 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.

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

where oracle O_{Res} is described in the experiment of signer ambiguity, $Query(\mathcal{A}, O_{\mathsf{Res}})$ is the set of queries made by $\mathcal{A}$ to oracle O_{Res}, and oracle O_{PSig} takes as input (M, PK_j) and outputs a

signature on M with respect to PK_A and PK_j generated using SK_A. 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.

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

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

where ASK^* is $\mathcal{A}$'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 Perfect Ambiguous Optimistic Fair Exchange

In a PAOFE scheme, we require that given a partial signature, no outsider should be able to learn any information about it. Specifically, the message on which the partial signature was generated, in addition to the identities of both the signer and the receiver should be completely hidden. To achieve this, we require that the verification algorithm in PAOFE to involve the secret key of the receiver, rather than the case that the partial signature is publicly verifiable in AOFE. Besides, we extend the resolution algorithm in AOFE to the resolution protocol in PAOFE. Since an algorithm can be seen as a non-interactive protocol, our model is more general and could capture a larger class of schemes.

Definition 3. *A perfect ambiguous optimistic fair exchange scheme involves the users (signers and verifiers) and the arbitrator, and consists of the following (probabilistic) polynomial-time algorithms/protocols:*

- **PMGen***: On input* 1^κ *where* κ *is a security parameter, this algorithm outputs a system parameter* PM*.*
- **Setup**$^{\mathsf{TTP}}$*: On input* PM*, the algorithm generates a secret key* ASK*, and a public key* APK *of the arbitrator.*
- **Setup**$^{\mathsf{User}}$*: On input* PM *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.*
- **Sig** *and* **Ver:** **Sig**$(M, \mathsf{SK}_i, \mathsf{PK}_i, \mathsf{PK}_j, \mathsf{APK})$*, outputs a (full) signature* σ *on message* M *of user* U_i *with the designated verifier* U_j*, while* **Ver**$(M, \sigma, \mathsf{PK}_i, \mathsf{PK}_j, \mathsf{APK})$ *outputs* $\top$ *or* $\perp$*, indicating* σ *is* U_i*'s valid full signature on* M *with the designated verifier* U_j *or not.*
- **PSig** *and* **PVer:** *These are partial signing and verification algorithms respectively.* **PSig**$(M, \mathsf{SK}_i, \mathsf{PK}_i, \mathsf{PK}_j, \mathsf{APK})$*, run by a signer* U_i*, outputs a partial signature* σ_P*, while* **PVer**$(M, \sigma_P, \mathsf{SK}_j, \mathsf{PK}_i, \mathsf{PK}_j, \mathsf{APK})$*, run by a verifier* U_j*, outputs* $\top$ *or* $\perp$*.*
- **Res:** *This is a resolution protocol between the verifier* U_j *and the arbitrator, involving a pair of interactive algorithms (*$\mathsf{Res}_\nu, \mathsf{Res}_\tau$*).* $\mathsf{Res}_\nu(M, \sigma_P, \mathsf{SK}_j, \mathsf{PK}_i, \mathsf{PK}_j, \mathsf{APK})$*, run by the verifier, outputs a full signature* σ*, or* $\perp$ *indicating the failure of resolving a partial signature.*

Resolution ambiguity property states that

- *any "resolved signature"* $\mathsf{Res}_\nu(M,$ **PSig**$(M, \mathsf{SK}_i, \mathsf{PK}_i, \mathsf{PK}_j, \mathsf{APK}), \mathsf{SK}_j, \mathsf{PK}_i, \mathsf{PK}_j, \mathsf{APK})$ *is computationally indistinguishable from the "actual signature"* **Sig**$(M, \mathsf{SK}_i, \mathsf{PK}_i, \mathsf{PK}_j, \mathsf{APK})$*.*

3.1 PAOFE models

– **Perfect ambiguity**: Intuitively, we require that no outsiders, even the arbitrator, should be able to learn any information about a partial signature such as the content of the message or the identities of the signer and receiver. This ensures the privacy for both the signer and the receiver. To achieve this property, we require that in the view of an outsider, the partial signature is indistinguishable to a signature randomly sampled from the signature space. Formally, we consider the following experiment in which $\mathcal{A}$ is a probabilistic polynomial-time distinguisher:

Experiment $\texttt{PAM}$:

$$\begin{aligned}
\mathsf{PM} &\leftarrow \mathbf{PMGen}(1^k)\\
(\mathsf{APK}, \mathsf{ASK}^*) &\leftarrow \mathcal{A}(\mathsf{PM})\\
(\mathsf{SK}_B, \mathsf{PK}_B) &\leftarrow \mathbf{Setup}^{\mathsf{User}}(\mathsf{PM}, \mathsf{APK})\\
(M, (\mathsf{SK}_A, \mathsf{PK}_A), \Upsilon) &\leftarrow \mathcal{A}^{O^B_{\mathbf{PSig}}, O^B_{\mathbf{FakePSig}}, O^B_{\mathbf{PVer}}}(\mathsf{ASK}^*, \mathsf{APK}, \mathsf{PK}_B)\\
b &\leftarrow \{0,1\}\\
\sigma_P &\leftarrow \begin{cases} \mathbf{PSig}(M, \mathsf{SK}_A, \mathsf{PK}_A, \mathsf{PK}_B, \mathsf{APK}) \text{ if } b = 0 \\ \sigma'_P \leftarrow \mathcal{S} \text{ if } b = 1 \end{cases}\\
b' &\leftarrow \mathcal{A}^{O^B_{\mathbf{PSig}}, O^B_{\mathbf{FakePSig}}, O^B_{\mathbf{PVer}}}(\sigma_P, \Upsilon)\\
\text{success of } \mathcal{A} &:= [b' = b\\
&\qquad \wedge (M, \sigma_P, \mathsf{PK}_A) \notin Query(\mathcal{A}, O^B_{\mathbf{PVer}})]
\end{aligned}$$

where Υ is $\mathcal{A}$'s state information, $\mathcal{S}$ is the whole partial signature space, oracle $O^B_{\mathbf{PSig}}$ takes as input (M, PK_j) and outputs a partial signature of PK_B's on M with the receiver's public key being PK_j, oracle $O^B_{\mathbf{FakePSig}}$ takes as input (M, PK_i) and returns a fake partial signature of user U_i's generated using SK_B on M with the receiver's public key being PK_B, oracle $O^B_{\mathbf{PVer}}$ takes as input a partial signature σ_P of user PK_i's on message M with the verifier being PK_B, i.e., $(M, \sigma_P, \mathsf{PK}_i)$, and outputs $\top$ or $\bot$, and $Query(\mathcal{A}, O^B_{\mathbf{PVer}})$ is the set of queries $\mathcal{A}$ issued to oracle $O^B_{\mathbf{PVer}}$. In all these oracle queries, $\mathcal{A}$ can arbitrarily choose a public key PK_i, probably without knowing the corresponding private key. However, as in AOFE, we do require that there exists a polynomial time algorithm to check the validity of the key pair output by $\mathcal{A}$, i.e., if SK_A matches PK_A, or if (SK_A, PK_A) is a possible output of $\mathbf{Setup}^{\mathsf{User}}$. Note that in previous ambiguous optimistic fair exchange models, the partial verification oracle $O^B_{\mathbf{PVer}}$ was not provided, as a partial signature is publicly verifiable. To cope with the change in PAOFE that partial signature is no longer publicly verifiable, we provide a partial signature verification oracle to the adversary in the security model.

– **Signer Ambiguity**: Informally, *signer ambiguity* means that given a partial signature σ_P from a signer A, a verifier B should not be able to convince others that σ_P was indeed generated by A, as B may forge partial signatures that look indistinguishable from those generated by A. Formally, we define an experiment in which $\mathcal{A}$ is a probabilistic polynomial-time distinguisher.

Experiment $\texttt{SAM}$:

$$\begin{aligned}
\mathsf{PM} &\leftarrow \mathbf{PMGen}(1^k)\\
(\mathsf{ASK}, \mathsf{APK}) &\leftarrow \mathbf{Setup}^{\mathsf{TTP}}(\mathsf{PM})\\
(M, (\mathsf{SK}_0, \mathsf{PK}_0), (\mathsf{SK}_1, \mathsf{PK}_1), \Upsilon) &\leftarrow \mathcal{A}^{O_{\mathbf{Res}}}(\mathsf{APK})\\
b &\leftarrow \{0,1\}\\
\sigma_P &\leftarrow \begin{cases} \mathbf{PSig}(M, \mathsf{SK}_0, \mathsf{PK}_0, \mathsf{PK}_1, \mathsf{APK}), \ b = 0 \\ \mathbf{FakePSig}(M, \mathsf{SK}_1, \mathsf{PK}_0, \mathsf{PK}_1, \mathsf{APK}), \ b = 1 \end{cases}\\
b' &\leftarrow \mathcal{A}^{O_{\mathbf{Res}}}(\sigma_P, \Upsilon)\\
\text{success of } \mathcal{A} &:= [b' = b\\
&\qquad \wedge\ (M, \mathsf{PK}_0, \mathsf{PK}_1) \notin Query(\mathcal{A}, O_{\mathbf{Res}})
\end{aligned}$$

where Υ is $\mathcal{A}$'s state information, oracle $O_{\mathbf{Res}}$ takes an input $(M, \mathsf{PK}_i, \mathsf{PK}_j)$ and starts an execution of the **Res** protocol with the adversary running the interactive algorithm $\mathsf{Res}_{\mathcal{R}}$, algorithm **FakePSig** is a fake partial signature signing algorithm and $\mathbf{FakeSig}(M, \mathsf{SK}_j, \mathsf{PK}_i, \mathsf{PK}_j, \mathsf{APK})$ outputs a forged partial signature σ_P on M of user U_i with the designated verifier U_j generated using SK_j, and $Query(\mathcal{A}, O_{\mathbf{Res}})$ is the set of queries $\mathcal{A}$ issued to the resolution oracle $O_{\mathbf{Res}}$. In this oracle query, $\mathcal{A}$ can arbitrarily choose two public key PK_i and PK_j probably without knowing the corresponding private key. However, as in the previous experiment, we do require that there exists a PPT algorithm to check the validity of the two key pairs output by $\mathcal{A}$, i.e., if SK_b matches PK_b for $b = 0, 1$, or if (SK_b, PK_b) is a possible output of $\mathbf{Setup}^{\mathsf{User}}$.

– **Security Against Signers**: We require that any PPT adversary $\mathcal{A}$, who models a dishonest signer, succeeds with at most negligible probability in the following experiment:

Experiment `SAS`:

$$\begin{aligned}
\mathsf{PM} &\leftarrow \mathbf{PMGen}(1^k)\\
(\mathsf{ASK}, \mathsf{APK}) &\leftarrow \mathbf{Setup}^{\mathsf{TTP}}(\mathsf{PM})\\
(\mathsf{SK}_B, \mathsf{PK}_B) &\leftarrow \mathbf{Setup}^{\mathsf{User}}(\mathsf{PM}, \mathsf{APK})\\
(M, \sigma_P, \mathsf{PK}_A) &\leftarrow \mathcal{A}^{O^B_{\mathbf{PSig}}, O^B_{\mathbf{FakePSig}}, O^B_{\mathbf{PVer}}, O_{\mathbf{Res}}}(\mathsf{APK}, \mathsf{PK}_B)\\
\mathsf{Input}_{\mathcal{T}} &:= (M, \mathsf{ASK}, \mathsf{PK}_A, \mathsf{PK}_B)\\
\mathsf{Input}_{\mathcal{V}} &:= (M, \sigma_P, \mathsf{SK}_B, \mathsf{PK}_A, \mathsf{PK}_B, \mathsf{APK})\\
[\mathsf{Res}_{\mathcal{T}}(\mathsf{Input}_{\mathcal{T}}) \rightarrow \mathsf{state}_{\mathcal{T}}] &\overset{\mathbf{Res}}{\Longleftrightarrow} [\mathsf{Res}_{\mathcal{V}}(\mathsf{Input}_{\mathcal{V}}) \rightarrow \sigma]\\
\text{success of } \mathcal{A} &:= [\mathbf{PVer}(M, \sigma_P, \mathsf{SK}_B, \mathsf{PK}_A, \mathsf{PK}_B, \mathsf{APK}) = \top\\
&\quad \wedge\ \mathbf{Ver}(M, \sigma, \mathsf{PK}_A, \mathsf{PK}_B, \mathsf{APK}) = \bot\\
&\quad \wedge\ (M, \mathsf{PK}_A) \notin Query(\mathcal{A}, O^B_{\mathbf{FakePSig}})]
\end{aligned}$$

where all the four oracles are described in the previous experiments, $Query(\mathcal{A}, O^B_{\mathbf{FakePSig}})$ is the set of queries made by $\mathcal{A}$ to oracle $O^B_{\mathbf{FakePSig}}$. Note that the adversary is not allowed to corrupt PK_B, otherwise it can easily success in the experiment by simply using SK_B to produce a fake partial signature under public keys PK_A, PK_B and outputting it.

– **Security Against Verifiers**: We require that any PPT adversary $\mathcal{A}$, who models a dishonest verifier, succeeds with at most negligible probability in the following experiment:

Experiment `SAV`:

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

where oracle $O_{\mathbf{Res}}$ is described in the previous experiments, oracle $O_{\mathbf{PSig}}$ takes as input (M, PK_j) and outputs a partial signature of PK_A's on M with the receiver's public key being PK_j generated using SK_A, oracle $O_{\mathbf{FakePSig}}$ takes as input (M, PK_i) and returns a fake partial signature of user U_i's generated using SK_A on M with the receiver's public key being PK_A, oracle $O_{\mathbf{PVer}}$ takes as input a partial signature σ_P of user U_i's on message M with the receiver's public key being PK_A, i.e., $(M, \sigma_P, \mathsf{PK}_i)$, and outputs $\top$ or $\bot$, and $Query(\mathcal{A}, O_{\mathbf{Res}})$ is the set of queries $\mathcal{A}$ issued to the resolution oracle. In the queries to the three oracles $O_{\mathbf{PSig}}$, $O_{\mathbf{FakePSig}}$ and $O_{\mathbf{PVer}}$, $\mathcal{A}$ can arbitrarily choose a public key PK_i, probably without knowing the corresponding private key.

– **Security Against the Arbitrator**: We require that any PPT adversary $\mathcal{A}$, who models a dishonest arbitrator, succeeds with at most negligible probability in the following experiment:

Experiment $\mathtt{SAA}$:

$$\begin{aligned}
\mathsf{PM} &\leftarrow \mathbf{PMGen}(1^k)\\
(\mathsf{APK}, \mathsf{ASK}^*) &\leftarrow \mathcal{A}(\mathsf{PM})\\
(\mathsf{SK}_A, \mathsf{PK}_A) &\leftarrow \mathbf{Setup}^{\mathsf{User}}(\mathsf{PM}, \mathsf{APK})\\
(M, \sigma, \mathsf{PK}_B) &\leftarrow \mathcal{A}^{O_{\mathbf{PSig}}, O_{\mathbf{FakePSig}}, O_{\mathbf{PVer}}}(\mathsf{ASK}^*, \mathsf{APK}, \mathsf{PK}_A)\\
\text{success of } \mathcal{A} &:= [\mathbf{Ver}(M, \sigma, \mathsf{PK}_A, \mathsf{PK}_B, \mathsf{APK}) = \top\\
&\qquad \wedge (M, \mathsf{PK}_B) \notin Query(\mathcal{A}, O_{\mathbf{PSig}})]
\end{aligned}$$

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

4 Generic Construction

In this section, we will present a generic construction of PAOFE. Let Γ = (PMGen, $\mathsf{Setup}^{\mathsf{TTP}}$, $\mathsf{Setup}^{\mathsf{User}}$, Sig, Ver, PSig, PVer, Res) be an ambiguous optimistic fair exchange scheme. Let $\mathcal{E}$ = (Kg, Enc, Dec) be a public key encryption scheme that is $\mathsf{IE\text{-}IK\text{-}CCA}$ secure.
A perfect ambiguous optimistic fair exchange can be constructed as follows:

– **PMGen**: This algorithm calls $\Gamma.\mathsf{PMGen}(1^\kappa) \rightarrow PM$ where κ is a security parameter, and outputs $\mathsf{PM} := PM$.
– **Setup**$^{\mathsf{TTP}}$: The arbitrator runs $\Gamma.\mathsf{Setup}^{\mathsf{TTP}}(\mathsf{PM}) \rightarrow (ASK, APK)$, and sets $(\mathsf{ASK}, \mathsf{APK}) := (ASK, APK)$.
– **Setup**$^{\mathsf{User}}$: Each user U_i runs $\Gamma.\mathsf{Setup}^{\mathsf{User}}(\mathsf{PM}, APK) \rightarrow (SK_i, PK_i)$ and $\mathcal{E}.\mathsf{Kg}(1^\kappa) \rightarrow (ek_i, dk_i)$ respectively, and sets $(\mathsf{SK}_i, \mathsf{PK}_i) := ((SK_i, dk_i), (PK_i, ek_i))$.
– **PSig**: To partially sign a message M with the verifier U_j, U_i runs $\Gamma.\mathsf{PSig}(M||\ \mathsf{PK}_i||\ \mathsf{PK}_j, SK_i, PK_i, PK_j, APK) \rightarrow \sigma'_P$ and then encrypts it under U_j's public encryption key ek_j by running $c = \mathcal{E}.\mathsf{Enc}_{ek_j}(\sigma'_P)$. The partial signature is set as $\sigma_P := c$.
– **PVer**: On receiving a partial signature σ_P on message M from the signer U_i, user U_j decrypts it using its own decryption key dk_j, i.e., $\sigma'_P = \mathcal{E}.\mathsf{Dec}_{dk_j}(\sigma_P)$, and then checks if $\Gamma.\mathsf{PVer}(M||\mathsf{PK}_i||\mathsf{PK}_j, \sigma'_P, PK_i, PK_j, APK) = \top$. If so, it accepts; otherwise, it rejects.
– **Sig**: To fully sign a message M for the verifier U_j, U_i calls $\Gamma.\mathsf{Sig}(M||\ \mathsf{PK}_i||\ \mathsf{PK}_j, SK_i, PK_i, PK_j, APK) \rightarrow \sigma$ and sends σ to U_j.
– **Ver**: On receiving a full signature σ from U_i, U_j outputs $\Gamma.\mathsf{Ver}(M||\ \mathsf{PK}_i||\ \mathsf{PK}_j, \sigma, PK_i, PK_j, APK)$.
– **Res**: Given a partial signature σ_P on message M from the signer U_i, user U_j decrypts it using its own decryption key dk_j, i.e., $\sigma'_P = \mathcal{E}.\mathsf{Dec}_{dk_j}(\sigma_P)$, and sends $(M, \sigma'_P, \mathsf{PK}_i, \mathsf{PK}_j)$ to the arbitrator.The arbitrator first checks the validity of σ'_P by running $\Gamma.\mathsf{PVer}(M||\ \mathsf{PK}_i||\ \mathsf{PK}_j, \sigma'_P, PK_i, PK_j, APK)$. If it's invalid, it returns $\perp$ to U_j. Otherwise, it returns $\Gamma.\mathsf{Res}(M||\ \mathsf{PK}_i||\ \mathsf{PK}_j, \sigma'_P, ASK, PK_i, PK_j)$ to U_j.

4.1 Security Analysis

Our generic construction is secure according to the model in Section 3.1. Detail security analysis is presented in Appendix A.

5 Conclusion

We proposed the notion of perfect ambiguous 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.

Our generic construction involves an encryption and an AOFE scheme and thus, it is bounded to be less efficient than AOFE. We leave it as our future work to construct more efficient PAOFE schemes, probably without directly using any encryption scheme.

References

1. N. Asokan, M. Schunter, and M. Waidner. Optimistic Protocols for Fair Exchange. In R. Graveman, P. A. Janson, C. Neumann, and L. Gong, editors, *ACM Conference on Computer and Communications Security*, pages 7–17. ACM, 1997.
2. N. Asokan, V. Shoup, and M. Waidner. Optimistic Fair Exchange of Digital Signatures (Extended Abstract). In K. Nyberg, editor, *EUROCRYPT*, volume 1403 of *Lecture Notes in Computer Science*, pages 591–606. Springer, 1998.
3. B. Barak, R. Canetti, J. B. Nielsen, and R. Pass. Universally Composable Protocols with Relaxed Set-Up Assumptions. In *FOCS*, pages 186–195. IEEE Computer Society, 2004.
4. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-Privacy in Public-Key Encryption. In C. Boyd, editor, *ASIACRYPT*, volume 2248 of *Lecture Notes in Computer Science*, pages 566–582. Springer, 2001.
5. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and Verifiably Encrypted Signatures from Bilinear Maps. In E. Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 416–432. Springer, 2003.
6. J. Camenisch and I. Damgård. Verifiable Encryption, Group Encryption, and Their Applications to Separable Group Signatures and Signature Sharing Schemes. In T. Okamoto, editor, *ASIACRYPT*, volume 1976 of *Lecture Notes in Computer Science*, pages 331–345. Springer, 2000.
7. R. Cramer and V. Shoup. A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack. In H. Krawczyk, editor, *CRYPTO*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer, 1998.
8. Y. Dodis, P. J. Lee, and D. H. Yum. Optimistic Fair Exchange in a Multi-user Setting. In T. Okamoto and X. Wang, editors, *Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 118–133. Springer, 2007.
9. Y. Dodis and L. Reyzin. Breaking and Repairing Optimistic Fair Exchange from PODC 2003. In M. Yung, editor, *Digital Rights Management Workshop*, pages 47–54. ACM, 2003.
10. P. D. Ezhilchelvan and S. K. Shrivastava. A Family of Trusted Third Party Based Fair-Exchange Protocols. *IEEE Trans. Dependable Sec. Comput.*, 2(4):273–286, 2005.
11. J. A. Garay, M. Jakobsson, and P. D. MacKenzie. Abuse-Free Optimistic Contract Signing. In M. J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 449–466. Springer, 1999.
12. S. Heidarvand and J. L. Villar. A Fair and Abuse-Free Contract Signing Protocol from Boneh-Boyen Signature. In J. Camenisch and C. Lambrinoudakis, editors, *EuroPKI*, volume 6711 of *Lecture Notes in Computer Science*, pages 125–140. Springer, 2010.
13. Q. Huang, D. S. Wong, and W. Susilo. Group-oriented Fair Exchange of Signatures. *Inf. Sci.*, 181(16):3267–3283, 2011.
14. Q. Huang, G. Yang, D. S. Wong, and W. Susilo. Ambiguous Optimistic Fair Exchange. In J. Pieprzyk, editor, *ASIACRYPT*, volume 5350 of *Lecture Notes in Computer Science*, pages 74–89. Springer, 2008.
15. Q. Huang, G. Yang, D. S. Wong, and W. Susilo. Efficient Optimistic Fair Exchange Secure in the Multi-user Setting and Chosen-Key Model without Random Oracles. In T. Malkin, editor, *CT-RSA*, volume 4964 of *Lecture Notes in Computer Science*, pages 106–120. Springer, 2008.
16. S. Lu, R. Ostrovsky, A. Sahai, H. Shacham, and B. Waters. Sequential Aggregate Signatures and Multisignatures Without Random Oracles. In S. Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 465–485. Springer, 2006.
17. S. Micali. Simple and Fast Optimistic Protocols for Fair Electronic Exchange. In E. Borowsky and S. Rajsbaum, editors, *PODC*, pages 12–19. ACM, 2003.
18. Y. Okada, Y. Manabe, and T. Okamoto. An Optimistic Fair Exchange Protocol and Its Security in the Universal Composability Framework. *IJACT*, 1(1):70–77, 2008.

19. M. Rückert and D. Schröder. Security of Verifiably Encrypted Signatures and a Construction without Random Oracles. In H. Shacham and B. Waters, editors, *Pairing*, volume 5671 of *Lecture Notes in Computer Science*, pages 17–34. Springer, 2009.
20. G. Wang. An Abuse-free Fair Contract Signing Protocol Based on the RSA Signature. In A. Ellis and T. Hagino, editors, *WWW*, pages 412–421. ACM, 2005.
21. J. Zhang and J. Mao. A Novel Verifiably Encrypted Signature Scheme Without Random Oracle. In E. Dawson and D. S. Wong, editors, *ISPEC*, volume 4464 of *Lecture Notes in Computer Science*, pages 65–78. Springer, 2007.

A Security Analysis

Obviously the resolution ambiguity property in PAOFE follows from that in AOFE.

Theorem 1. *The generic construction is perfect ambiguous if* $\mathcal{E} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ *is an* IE-IK-CCA *secure encryption.*

Proof. To show perfect ambiguity, we convert any adversary $\mathcal{A}$ that wins the experiment PAM into an adversary $\mathcal{A}'$ that breaks the IE-IK-CCA security of $\mathcal{E}$. Recall that $\mathcal{A}'$ gets ek as input and has access to oracle O_{Dec}. Suppose the public parameter PM is generated. $\mathcal{A}$ first chooses a public adjudication key APK and outputs it, and keeps a corresponding secret state information ASK^* private. $\mathcal{A}'$ sets $PM := \mathsf{PM}$, $APK := \mathsf{APK}$, and runs $\Gamma.\mathsf{Setup}^{\mathsf{User}}(PM, APK) \rightarrow (SK_B, PK_B)$ and invokes $\mathcal{A}$ on input $\mathsf{PK}_B := (PK_B, ek)$.

Given a partial signature signing query $(M, \mathsf{PK}_j = (PK_j, ek_j))$ to oracle $O^B_{\mathbf{PSig}}$, $\mathcal{A}'$ runs $\Gamma.\mathsf{PSig}$ $(M||\mathsf{PK}_B||\mathsf{PK}_j, SK_B, PK_B, PK_j, APK) \rightarrow \sigma'_P$, and then encrypts σ'_P under ek_j by running $c = \mathcal{E}.\mathsf{Enc}_{ek_j}(\sigma'_P)$. $\mathcal{A}'$ returns c to $\mathcal{A}$ as the answer.

Given a fake partial signing query (M, PK_i) where $\mathsf{PK}_i = (PK_i, ek_i)$ to oracle $O^B_{\mathbf{FakePSig}}$, $\mathcal{A}'$ runs $\Gamma.\mathsf{PSig}(M||\mathsf{PK}_i||\mathsf{PK}_B, SK_B, PK_B, PK_i, APK) \rightarrow \sigma'_P$, and then encrypts σ'_P under ek_B by running $c = \mathcal{E}.\mathsf{Enc}_{ek}(\sigma'_P)$. $\mathcal{A}'$ returns c to $\mathcal{A}$ as the answer.

Given a partial signature verification query $(M, \sigma_P, \mathsf{PK}_i = (PK_i, ek_i))$ to $O^B_{\mathbf{PVer}}$, $\mathcal{A}'$ makes a decryption query σ_P to its own oracle O_{Dec}. Denote the answer from O_{Dec} is σ'_P. $\mathcal{A}'$ returns $\Gamma.\mathsf{PVer}$ $(M||\mathsf{PK}_i||\mathsf{PK}_B, \sigma'_P, PK_i, PK_B, APK)$ to $\mathcal{A}$.

At some time, $\mathcal{A}$ submits $(M^*, (\mathsf{SK}_A, \mathsf{PK}_A))$, where $\mathsf{SK}_A := (SK_A, dk_A)$, and SK_A matches PK_A. $\mathcal{A}'$ runs $\Gamma.\mathsf{PSig}(M^*||\mathsf{PK}_A||\mathsf{PK}_B, SK_A, PK_A, PK_B, APK) \rightarrow \sigma'_P$, and submits σ'_P to its own challenger, which returns a ciphertext c^*. $\mathcal{A}'$ forwards $\sigma_P := c^*$ to $\mathcal{A}$, and then continues to simulate the oracles $O^B_{\mathbf{PSig}}$ and $O^B_{\mathbf{FakePSig}}$ in the same way as above. About the further queries $(M, \sigma_p, \mathsf{PK}_i)$ where $\mathsf{PK}_i = (PK_i, ek_i)$ to oracle $O^B_{\mathbf{PVer}}$, we distinguish the following two case:

1. $\sigma_P \neq c^*$. In this case, $\mathcal{A}'$ simulates $O^B_{\mathbf{PVer}}$ in the same way as above.
2. $\sigma_P = c^*$. In this case, $\mathcal{A}'$ just returns $\perp$ to the adversary A. First of all, we can exclude the subcase where $(M, c^*, \mathsf{PK}_i) = (M^*, c^*, \mathsf{PK}_A)$, because $(M^*, c^*, \mathsf{PK}_A)$ is prohibited from being queried to oracle $O^B_{\mathbf{PVer}}$. If $(M, c^*, \mathsf{PK}_i) \neq (M^*, c^*, \mathsf{PK}_A)$ and $\mathbf{PVer}(M, c^*, \mathsf{SK}_B, \mathsf{PK}_i, \mathsf{PK}_B, \mathsf{APK}) = \top$, the probability of this happening is negligible.

It can be seen that the oracles $O^B_{\mathbf{PSig}}$, $O^B_{\mathbf{FakePSig}}$ and $O^B_{\mathbf{PVer}}$ are simulated properly by $\mathcal{A}'$. Finally, $\mathcal{A}$ outputs a bit d. $\mathcal{A}'$ outputs a bit $b' = d$, and A' has never issued a query to its decryption oracle O_{Dec} on input c^*. If $\mathcal{A}$ succeeds in the experiment, $\mathcal{A}'$ also succeeds in outputting the bit b'. Therefore $\mathcal{A}'$'s advantage is also non-negligible. □

Theorem 2. *The generic construction is signer ambiguous if* $\Gamma = (\mathsf{PMGen}, \mathsf{Setup}^{\mathsf{TTP}}, \mathsf{Setup}^{\mathsf{User}}, \mathsf{Sig}, \mathsf{Ver}, \mathsf{PSig}, \mathsf{PVer}, \mathsf{Res})$ *is signer ambiguous.*

Proof. To show signer ambiguity, we convert any adversary $\mathcal{A}$ that wins the experiment SAM into an adversary $\mathcal{A}'$ that breaks the signer ambiguity security of Γ. Recall that $\mathcal{A}'$ gets APK as input and has access to oracle O'_{Res}. Suppose the public parameter PM is generated. $\mathcal{A}'$ invokes $\mathcal{A}$ on input $\mathsf{APK} := APK$.

Given a resolution query $(M, \mathsf{PK}_i, \mathsf{PK}_j)$ where $\mathsf{PK}_i = (PK_i, ek_i)$ and $\mathsf{PK}_j = (PK_j, ek_j)$ to $O_{\mathbf{Res}}$, suppose $\mathcal{A}$ sends σ'_P to $\mathcal{A}'$ in the first run of the protocol. $\mathcal{A}'$ makes a query $(M||\mathsf{PK}_i||\mathsf{PK}_j, \sigma'_P, PK_i, PK_j)$ to its own oracle O'_{Res} and returns the answer to $\mathcal{A}$.

At some time, $\mathcal{A}$ submits $(M^*, (\mathsf{SK}_0, \mathsf{PK}_0), (\mathsf{SK}_1, \mathsf{PK}_1))$, where $\mathsf{SK}_0 := (SK_0, dk_0)$, $\mathsf{PK}_0 := (PK_0, ek_0)$, $\mathsf{SK}_1 := (SK_1, dk_1)$, $\mathsf{PK}_1 := (PK_1, ek_1)$ and SK_b matches PK_b for $b = 0, 1$. $\mathcal{A}'$ submits $(M^*||\mathsf{PK}_0||\mathsf{PK}_1, (SK_0, PK_0), (SK_1, PK_1))$ to its own challenger, which returns a partial signature σ^*_P with respect to the secret key SK_b for some random choice $b \in \{0, 1\}$. $\mathcal{A}'$ encrypts σ^*_P under the public encryption key ek_1, i.e., $c = \mathcal{E}.\mathsf{Enc}_{ek_1}(\sigma^*_P)$, and forwards $\sigma_P := c^*$ to $\mathcal{A}$ as the answer. $\mathcal{A}'$ then continues to simulate the oracle $O_{\mathbf{Res}}$ in the same way as above.

It can be seen that the oracle $O_{\mathbf{Res}}$ is simulated properly by $\mathcal{A}'$. Finally, $\mathcal{A}$ outputs a bit d. $\mathcal{A}'$ outputs a bit $b' = d$ and halts. Since $\mathcal{A}$ is not allowed to issue a query $(M^*, \mathsf{PK}_0, \mathsf{PK}_1)$ to the resolution oracle $O_{\mathbf{Res}}$, $\mathcal{A}'$ has never made a query to its oracle O'_{Res} with respect to message $M^*||\mathsf{PK}_0||\mathsf{PK}_1$. If $\mathcal{A}$ succeeds in the experiment, $\mathcal{A}'$ also succeeds in outputting the bit b' with the same probability. $\mathcal{A}'$'s advantage is also non-negligible. □

Theorem 3. *The generic construction is secure against signers if* $\Gamma = (\mathsf{PMGen}, \mathsf{Setup}^{\mathsf{TTP}}, \mathsf{Setup}^{\mathsf{User}}, \mathsf{Sig}, \mathsf{Ver}, \mathsf{PSig}, \mathsf{PVer}, \mathsf{Res})$ *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 Γ. Recall that $\mathcal{A}'$ gets (APK, PK_B) as input and has access to oracles O'^{B}_{PSig} and O'_{Res}. $\mathcal{A}'$ runs $\mathcal{E}.\mathsf{KGen}(1^\kappa) \to (ek_B, dk_B)$ and invokes $\mathcal{A}$ on input $\mathsf{APK} := APK$ and $\mathsf{PK}_B := (PK_B, ek_B)$.

Given a partial signing query (M, PK_j) where $\mathsf{PK}_j = (PK_j, ek_j)$ to oracle $O^B_{\mathbf{PSig}}$, $\mathcal{A}'$ makes a query $(M||\mathsf{PK}_B||\mathsf{PK}_j, PK_j)$ to its own oracle O'^{B}_{PSig}. Denote the answer from O'^{B}_{PSig} is σ'_P. $\mathcal{A}'$ then encrypt it under the public encryption key ek_j, i.e., $c = \mathcal{E}.\mathsf{Enc}_{ek_j}(\sigma'_P)$, and returns $\sigma_P := c$ to $\mathcal{A}$ as the answer.

Given a fake partial signing query (M, PK_i) where $\mathsf{PK}_i = (PK_i, ek_i)$ to oracle $O^B_{\mathbf{FakePSig}}$, $\mathcal{A}'$ makes a query $(M||\mathsf{PK}_i||\mathsf{PK}_B, PK_i)$ to its own oracle O'^{B}_{PSig}. Denote the answer from O'^{B}_{PSig} is σ'_P. $\mathcal{A}'$ then encrypt it under the public encryption key ek_B, i.e., $c = \mathcal{E}.\mathsf{Enc}_{ek_B}(\sigma'_P)$, and returns $\sigma_P := c$ to $\mathcal{A}$ as the answer.

Given a partial signature verification query $(M, \sigma_P, \mathsf{PK}_i = (PK_i, ek_i))$ to $O^B_{\mathbf{PVer}}$, $\mathcal{A}'$ decrypts σ_P using its own decryption key dk_B, i.e., $\sigma'_P = \mathcal{E}.\mathsf{Dec}_{dk_B}(\sigma_P)$, and returns $\Gamma.\mathsf{PVer}(M||\mathsf{PK}_i||\mathsf{PK}_B, \sigma'_P, PK_i, PK_B, APK)$ to $\mathcal{A}$.

Given a resolution query $(M, \mathsf{PK}_i, \mathsf{PK}_j)$ where $\mathsf{PK}_i = (PK_i, ek_i)$ and $\mathsf{PK}_j = (PK_j, ek_j)$ to $O^B_{\mathbf{Res}}$, suppose $\mathcal{A}$ sends σ'_P to $\mathcal{A}'$ in the first run of the protocol. $\mathcal{A}'$ makes a query $(M||\mathsf{PK}_i||\mathsf{PK}_j, \sigma'_P, PK_i, PK_j)$ to its own oracle O'_{Res} and returns the answer to $\mathcal{A}$.

It can be seen that the oracles $O_{\mathbf{PSig}}$, $O^B_{\mathbf{FakePSig}}$, $O^B_{\mathbf{PVer}}$ and $O_{\mathbf{Res}}$ are simulated properly by $\mathcal{A}'$. Finally, $\mathcal{A}$ outputs a partial signature σ^*_P on message M^* under PK_A, PK_B where $\mathsf{PK}_A = (PK_A, ek_A)$. $\mathcal{A}'$ decrypts σ^*_P under the decryption key dk_B, i.e., $\tilde{\sigma_P} = \mathcal{E}.\mathsf{Dec}_{dk_B}(\sigma^*_P)$ and outputs $(M^*||\mathsf{PK}_A||\mathsf{PK}_B, \tilde{\sigma_P}, PK_A)$. Notice that $\tilde{\sigma_P}$ is a valid partial signature on message $M^*||\mathsf{PK}_A||\mathsf{PK}_B$ under PK_A and PK_B, i.e. $\Gamma.\mathsf{PVer}(M^*||\mathsf{PK}_A||\mathsf{PK}_B, \tilde{\sigma_P}, PK_A, PK_B, APK) = \top$, but it can not be resolved to a valid full signature by the resolution algorithm $\Gamma.\mathsf{Res}$, i.e. $\Gamma.\mathsf{Res}(M^*||\mathsf{PK}_A||\mathsf{PK}_B, \tilde{\sigma_P}, ASK, PK_A, PK_B) = \perp$. Since $\mathcal{A}$ is prohibited from making a query (M^*, PK_A) to oracle $O^B_{\mathbf{FakePSig}}$, $\mathcal{A}'$ has never issued a query $(M^*||\mathsf{PK}_A||\mathsf{PK}_B, PK_A)$ to its own oracle O'^{B}_{PSig}, thus if $\mathcal{A}$ succeeds in the experiment, $\mathcal{A}'$ also succeeds with the same probability in breaking the security against signers of Γ. □

Theorem 4. *The generic construction is secure against verifiers if* $\Gamma = (\mathsf{PMGen}, \mathsf{Setup}^{\mathsf{TTP}}, \mathsf{Setup}^{\mathsf{User}}, \mathsf{Sig}, \mathsf{Ver}, \mathsf{PSig}, \mathsf{PVer}, \mathsf{Res})$ *is secure against verifiers.*

Proof. To show security against verifiers, we convert any adversary $\mathcal{A}$ that wins the experiment $\mathtt{SAV}$ into an adversary $\mathcal{A}'$ that breaks the security against verifiers of Γ. Recall that $\mathcal{A}'$ gets (APK, PK_A) as input and has access to oracles O'_{PSig} and O'_{Res}. $\mathcal{A}'$ runs $\mathcal{E}.\mathsf{KGen}(1^\kappa) \to (ek_A, dk_A)$ and invokes $\mathcal{A}$ on input $\mathsf{APK} := APK$ and $\mathsf{PK}_A := (PK_A, ek_A)$.

Given a partial signing query (M, PK_j) where $\mathsf{PK}_j = (PK_j, ek_j)$ to oracle $O_{\mathbf{PSig}}$, $\mathcal{A}'$ makes a query $(M||\mathsf{PK}_A||\mathsf{PK}_j, PK_j)$ to its own oracle O'_{PSig}. Denote the answer from O'_{PSig} is σ'_P. $\mathcal{A}'$ then encrypt it under the public encryption key ek_j, i.e., $c = \mathcal{E}.\mathsf{Enc}_{ek_j}(\sigma'_P)$, and returns $\sigma_P := c$ to $\mathcal{A}$ as the answer.

Given a fake partial signing query (M, PK_i) where $\mathsf{PK}_i = (PK_i, ek_i)$ to oracle $O_{\mathbf{FakePSig}}$, $\mathcal{A}'$ makes a query $(M||\mathsf{PK}_i||\mathsf{PK}_A, PK_i)$ to its own oracle O'_{PSig}. Denote the answer from O'_{PSig} is σ'_P. $\mathcal{A}'$ then encrypt it under the public encryption key ek_A, i.e., $c = \mathcal{E}.\mathsf{Enc}_{ek_A}(\sigma'_P)$, and returns $\sigma_P := c$ to $\mathcal{A}$ as the answer.

Given a partial signature verification query $(M, \sigma_P, \mathsf{PK}_i = (PK_i, ek_i))$ to oracle $O_{\mathbf{PVer}}$, $\mathcal{A}'$ decrypts σ_P using its own decryption key dk_A, i.e., $\sigma'_P = \mathcal{E}.\mathsf{Dec}_{dk_A}(\sigma_P)$, and returns $\Gamma.\mathsf{PVer}$ $(M||\mathsf{PK}_i||\mathsf{PK}_A, \sigma'_P, PK_i, PK_A, APK)$ to $\mathcal{A}$.

Given a resolution query $(M, \mathsf{PK}_i, \mathsf{PK}_j)$ where $\mathsf{PK}_i = (PK_i, ek_i)$ and $\mathsf{PK}_j = (PK_j, ek_j)$ to $O_{\mathbf{Res}}$, suppose $\mathcal{A}$ sends σ'_P to $\mathcal{A}'$ in the first run of the protocol. $\mathcal{A}'$ makes a query $(M||\mathsf{PK}_i||\mathsf{PK}_j, \sigma'_P, PK_i, PK_j)$ to its own oracle O'_{Res} and returns the answer to $\mathcal{A}$.

It can be seen that the oracles $O_{\mathbf{FakePSig}}$, $O_{\mathbf{PSig}}$, $O_{\mathbf{PVer}}$ and $O_{\mathbf{Res}}$ are simulated properly by $\mathcal{A}'$. Finally, $\mathcal{A}$ returns a full signature σ^* on message M^* under PK_A, PK_B where $\mathsf{PK}_B = (PK_B, ek_B)$ such that $\mathbf{Ver}(M^*, \sigma^*, \mathsf{PK}_A, \mathsf{PK}_B, \mathsf{APK}) = \top$, which means $\Gamma.\mathsf{Ver}(M^*||\mathsf{PK}_A||\mathsf{PK}_B, \sigma^*, PK_A, PK_B, APK) = \top$. $\mathcal{A}'$ outputs $(M^*||\mathsf{PK}_A||\mathsf{PK}_B, \sigma^*, PK_B)$ and aborts. Since $\mathcal{A}$ is prohibited from making a query $(M^*, \mathsf{PK}_A, \mathsf{PK}_B)$ to oracle $O_{\mathbf{Res}}$, $\mathcal{A}'$ has never made a query with respect to message $M^*||\mathsf{PK}_A||\mathsf{PK}_B$ to its own oracle O'_{Res}. If $\mathcal{A}$ succeeds in the experiment, $\mathcal{A}'$ also succeeds in breaking the security against verifiers of Γ. Thus $\mathcal{A}'$'s advantage is also non-negligible. □

Theorem 5. *The generic construction is secure against the arbitrator if* $\Gamma = (\mathsf{PMGen}, \mathsf{Setup}^{\mathsf{TTP}}, \mathsf{Setup}^{\mathsf{User}}, \mathsf{Sig}, \mathsf{Ver}, \mathsf{PSig}, \mathsf{PVer}, \mathsf{Res})$ *is secure against the arbitrator.*

Proof. To show security against the arbitrator, we convert any adversary $\mathcal{A}$ that wins the experiment $\mathtt{SAA}$ into an adversary $\mathcal{A}'$ that breaks the security against the arbitrator of Γ. Suppose the public parameter PM is generated. $\mathcal{A}$ first chooses a public adjudication key APK and outputs it, and 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 $\mathcal{E}.\mathsf{KGen}(1^\kappa) \to (ek_A, dk_A)$ and invokes $\mathcal{A}$ on input $\mathsf{PK}_A := (PK_A, ek_A)$.

Given a partial signing query (M, PK_j) where $\mathsf{PK}_j = (PK_j, ek_j)$ to oracle $O_{\mathbf{PSig}}$, $\mathcal{A}'$ makes a query $(M||\mathsf{PK}_A||\mathsf{PK}_j, PK_j)$ to its own oracle O'_{PSig}. Suppose the answer from O'_{PSig} is σ'_P. $\mathcal{A}'$ then encrypt it under the public encryption key ek_j, i.e., $c = \mathcal{E}.\mathsf{Enc}_{ek_j}(\sigma'_P)$, and returns $\sigma_P := c$ to $\mathcal{A}$ as the answer.

Given a fake partial signing query (M, PK_i) where $\mathsf{PK}_i = (PK_i, ek_i)$ to oracle $O_{\mathbf{FakePSig}}$, $\mathcal{A}'$ makes a query $(M||\mathsf{PK}_i||\mathsf{PK}_A, PK_i)$ to its own oracle O'_{PSig}. Denote the answer from O'_{PSig} is σ'_P. $\mathcal{A}'$ then encrypt it under the public encryption key ek_A, i.e., $c = \mathcal{E}.\mathsf{Enc}_{ek_A}(\sigma'_P)$, and returns $\sigma_P := c$ to $\mathcal{A}$ as the answer.

Given a partial signature verification query $(M, \sigma_P, \mathsf{PK}_i = (PK_i, ek_i))$ to oracle $O_{\mathbf{PVer}}$, $\mathcal{A}'$ decrypts σ_P using its own decryption key dk_A, i.e., $\sigma'_P = \mathcal{E}.\mathsf{Dec}_{dk_A}(\sigma_P)$, and returns $\Gamma.\mathsf{PVer}$ $(M||\mathsf{PK}_i||\mathsf{PK}_A, \sigma'_P, PK_i, PK_A, APK)$ to $\mathcal{A}$.

It can be seen that the oracles $O_{\mathbf{FakePSig}}$, $O_{\mathbf{PSig}}$ and $O_{\mathbf{PVer}}$ are simulated properly by $\mathcal{A}'$. Finally, $\mathcal{A}$ returns a full signature σ^* on message M^* under PK_A, PK_B where $\mathsf{PK}_B = (PK_B, ek_B)$, such that $\mathbf{Ver}(M^*, \sigma^*, \mathsf{PK}_A, \mathsf{PK}_B, \mathsf{APK}) = \top$, which means $\Gamma.\mathsf{Ver}(M^*||\ \mathsf{PK}_A||\ \mathsf{PK}_B,\ \sigma^*,\ PK_A,\ PK_B,\ APK) = \top$. $\mathcal{A}'$ outputs $(M^*||\mathsf{PK}_A||\mathsf{PK}_B, \sigma^*, PK_B)$ and aborts. Since $\mathcal{A}$ is prohibited from making a query $(M^*, \mathsf{PK}_A, \mathsf{PK}_B)$ to oracle $O_{\mathbf{PSig}}$, $\mathcal{A}'$ has not made a query with respect to message $M^*||\mathsf{PK}_A||\mathsf{PK}_B$ to its own oracle O'_{PSig}. If $\mathcal{A}$ succeeds in the experiment, $\mathcal{A}'$ also succeeds with the same probability in breaking the security against the arbitrator of Γ, as σ^* is a valid full signature on message $M^*||\mathsf{PK}_A||\mathsf{PK}_B$ under PK_A, PK_B. Thus $\mathcal{A}'$'s advantage is also non-negligible. □