2013

Fairness in concurrent signatures revisited

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

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

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

Duncan S. Wong
City University of Hong Kong, dwong@uow.edu.au

Publication Details

Susilo, W., Au, M. Ho., Wang, Y. & Wong, D. S. (2013). Fairness in concurrent signatures revisited. Lecture Notes in Computer Science, 7959 318-329.

Fairness in concurrent signatures revisited

Abstract

Concurrent signature, introduced by Chen, Kudla and Paterson, is known to just fall short to solve the long standing fair exchange of signature problem without requiring any trusted third party (TTP). The price for not requiring any TTP is that the initial signer is always having some advantage over the matching signer in controlling whether the protocol completes or not, and hence, whether the two ambiguous signatures will bind concurrently to their true signers or not. In this paper, we examine the notion and classify the advantages of the initial signer into three levels, some of which but not all of them may be known in the literature. - Advantage level 0 is the commonly acknowledged fact that concurrent signature is not abuse-free since an initial signer who holds a keystone can always choose to complete or abort a concurrent signature protocol run by deciding whether to release the keystone or not. - Advantage level 1 refers to the fact that the initial signer can convince a third party that both ambiguous signatures are valid without actually making the signatures publicly verifiable. - Advantage level 2 allows the initial signer to convince a third party that the matching signer agrees to commit to a specific message, and nothing else.We stress that advantage level 2 is not about proving the possession of a keystone. Proving the knowledge of a keystone would make the malicious initial signer accountable as this could only be done by the initial signer. We remark that the original security models for concurrent signature do not rule out the aforementioned advantages of the initial signer. Indeed, we show that theoretically, the initial signer always enjoys the above advantages for any concurrent signatures. Our work demonstrates a clear gap between the notion of concurrent signature and optimistic fair exchange (OFE) in which no party enjoys advantage level 1. Furthermore, in a variant known as Ambiguous OFE, no party enjoys advantage level 1 and 2.

Keywords

revisited, signatures, concurrent, fairness

Disciplines

Engineering | Science and Technology Studies

Publication Details

Susilo, W., Au, M. Ho., Wang, Y. & Wong, D. S. (2013). Fairness in concurrent signatures revisited. Lecture Notes in Computer Science, 7959 318-329.

Fairness in Concurrent Signatures Revisited

Willy Susilo[1]*, Man Ho Au[1], Yang Wang[1] and Duncan S. Wong[2]

[1]Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Australia
Email: {wsusilo, aau}@uow.edu.au
yw990@uowmail.uow.edu.au
[2]City University of Hong Kong, Hong Kong
Email: duncan@cityu.edu.hk

Abstract. Concurrent signature, introduced by Chen, Kudla and Paterson, is known to just fall short to solve the long standing fair exchange of signature problem without requiring any trusted third party (TTP). The price for not requiring any TTP is that the initial signer is always having some advantage over the matching signer in controlling whether the protocol completes or not, and hence, whether the two ambiguous signatures will bind concurrently to their true signers or not. In this paper, we examine the notion and classify the advantages of the initial signer into three levels, some of which but not all of them may be known in the literature.

- Advantage level 0 is the commonly acknowledged fact that concurrent signature is not abuse-free since an initial signer who holds a keystone can always choose to complete or abort a concurrent signature protocol run by deciding whether to release the keystone or not.
- Advantage level 1 refers to the fact that the initial signer can convince a third party that both ambiguous signatures are valid without actually making the signatures publicly verifiable.
- Advantage level 2 allows the initial signer to convince a third party that the matching signer agrees to commit to a specific message, and nothing else. We stress that advantage level 2 is *not* about proving the possession of a keystone. Proving the knowledge of a keystone would make the malicious initial signer accountable as this could only be done by the initial signer.

We remark that the original security models for concurrent signature do not rule out the aforementioned advantages of the initial signer. Indeed, we show that theoretically, the initial signer always enjoys the above advantages for any concurrent signatures. Our work demonstrates a clear gap between the notion of concurrent signature and optimistic fair exchange (OFE) in which no party enjoys advantage level 1. Furthermore, in a variant known as Ambiguous OFE, no party enjoys advantage level 1 and 2.

Keywords: concurrent signatures, scenarios, fairness

1 Introduction

Fair exchange of digital signatures has been considered as a fundamental problem in cryptography. It is a useful cryptographic protocol that allows secure and fair e-commerce applications to exchange digital signatures for legal contracts or agreements. Nowadays, goods and services are being exchanged electronically over the Internet. Our main goal is to ensure that the exchanges are *fair*, which means that at the end of an exchange between two parties, either both parties receive the complete items or none of them obtains anything.

Fairness in exchanging signatures is normally achieved with the help of a trusted third party (TTP), which is often offline. There were several attempts where a fair exchange of signatures can be achieved with a "semi-trusted" TTP who can be called upon to handle disputes between

* W. Susilo is supported by ARC Future Fellowship FT0991397.

signers [2]. This type of fair exchanges is also referred to as an optimistic fair exchange (OFE) [3,4]. A well-known open problem in fair exchange is the requirement of a dispute resolving TTP whose role cannot be replaced by a normal certification authority.

In Eurocrypt 2004 [6], Chen, Kudla and Paterson presented a new cryptographic primitive called "concurrent signature" to allow two parties to produce two ambiguous signatures, such that both signatures do not bind to their true signers. Upon the release of the signatures, any third party cannot identify the true signer who generated the signature. However, upon the release of an extra piece of information called the keystone, both signatures will bind concurrently. The merit of concurrent signature relies on the fact that there is *no* TTP required. This has been considered to be very practical, and hence, concurrent signatures have been promising to be adopted in practice (c.f. OFE where the need for a TTP may not exist in the real scenario). The demand of requiring a TTP in cryptographic schemes has made cryptographic schemes less attractive for adoption in practice, and hence, concurrent signatures have been very promising in bridging the gap between theoreticians and practitioners. Further, Chen, Kudla and Paterson presented a concrete concurrent signature scheme based on a variant of Schnorr based ring signature scheme [1]. In their scheme, any third party cannot be convinced that a signature has indeed been signed by one particular signer, since any signer can always generate this signature by himself/herself.

In a concurrent signature protocol run, there are two parties involved, namely Alice and Bob (or A and B, respectively). Since one party is required to create a keystone and sends the first message to the other party, we call this party as the *initial signer*. A party who responds to the initial signature by creating another signature is called a *matching signer*. Without losing generality, throughout this paper, we assume that Alice (or A, resp.) is the initial signer and Bob (or B, resp.) is the matching signer.

It is acknowledged that concurrent signature is not perfectly fair, since Alice is in control of the time at which the keystone is released and thus, control *when* the ambiguous signatures become binding to their respective signers. Alice can even decide not to complete a concurrent signature protocol run if Alice decided not to to release the keystone ultimately. Nonetheless, the concept of *fairness* in a concurrent signature is defined as follows: "once Alice releases the keystone, both ambiguous signatures from Alice and Bob binds concurrently". Specifically, it is required that Alice cannot output a maliciously crafted keystone so that the ambiguous signature from Bob together with this keystone passes the verification algorithm, yet verification of the ambiguous signature created by Alice (perhaps also maliciously) together with this keystone would output failure. We call this definition of fairness a "white-box" guarantee, as the malicious Alice is required to convert the ambiguous signature corresponding to Bob to a "well-formed" publicly verifiable signature under Bob to be considered successful. This is possibly sufficient for legal contract signing purpose, since the contract is only valid if and only if a "well-formed" signature is present. This definition is adopted in Chen et al.'s paper and the subsequent works.

1.1 Fairness in Practice

In this paper, we make the observation that the formal definition of fairness does *not* necessarily capture the fairness in practice completely. For example, there is *no guarantee* that Alice could not convince a third party Carol that Bob has signed a message, M_B, that is, committed to M_B, *without* revealing the keystone. We identify that a malicious initial signer may enjoy three levels of advantages in concurrent signature.

(a) Level 0 advantage is inherent in concurrent signature. The initial signer can always choose to abort or complete a concurrent signature protocol run.
(b) Level 1 advantage allows the initial signer to demonstrate the fact that he/she is capable of making both signatures valid if he/she wanted to, without actually making both signatures publicly verifiable.
(c) Level 2 advantage allows the initial signer to convince a third party that the matching signer has agreed to committed to a certain message, for example, M_B, without revealing anything else.

These advantage levels have concrete implications regarding the use of concurrent signatures in practical scenarios. As discussed by Chen et al. in their seminal paper [6], it is very often the case that the matching signer would not mind sacrificing level 0 advantage. However, regarding level 1 and 2 advantages, concurrent signatures may not be suitable in some other scenarios such as tendering systems (c.f. [6] as we will show with details in the later part of this paper). On the other hand, contrary to the common belief that concurrent signature is applicable to tendering systems (such as [6,17,18]), level 1 advantage to a malicious initial signer could be unacceptable to some of the suppliers. Hence, in those scenarios, the OFE [2–4,7] or Ambiguous OFE [10–12] systems are indeed more suitable compared to concurrent signatures.

1.2 Our Contributions

Firstly, we show that any constructions of concurrent signature following Chen, Kudla and Peterson is *always* subject to abuse by the initial signer, with advantage level 1 and 2, in addition to the commonly acknowledged advantage level 0. We present generic methods that allow a malicious initial signer to convince any third party that he/she has the ability to make both signatures verifiable (level 1), or that the matching signer has committed to his message (level 2). Secondly, we examine one variant of concurrent signatures, namely, asymmetric concurrent signature [13] and demonstrate how a malicious initial signer can exhibit his/her level 1 advantage in an effective manner. Our attack is practical and its implication may discourage the adoption of concurrent signatures in some application scenarios, and particularly, when it is undesirable to allow a malicious initial signer to convince anyone of the binding of the matching signer's signature without making the signature publicly verifiable.

1.3 Related Work

Following the seminal work by Chen et al., many subsequent concurrent signature schemes have been proposed (such as [13,14,16,17]). Nguyen [13] proposed an interesting variant that embraces the asymmetric property of concurrent signatures (c.f. the symmetric property of all the previously known concurrent signature schemes). Furthermore, Nguyen noted that the construction techniques of an asymmetric concurrent signature scheme can be used for constructing a multi-party concurrent signature scheme, which is the solution to the open problem stated in Chen et al.'s seminal work [6]. Subsequently, Tonien, Susilo and Safavi-Naini [16] proposed a multi-party concurrent signature scheme that uses a different model from the construction achieved from [13].

In an orthogonal direction, Susilo, Mu and Zhang [14] investigated the privacy issue in concurrent signatures. They observed that prior the release of a keystone, and just from the two ambiguous signatures, any third party can already conclude that the two ambiguous signatures must be created by these two possible signers. At the same time, if the possible signers are believed to be honest, the outsider can already tell who the actual signer is corresponding to each ambiguous signature. They then introduced a stronger requirement called *perfect ambiguity*, which requires the ambiguous signatures to remain "ambiguous" even if the two potential signers are known to be honest. Unfortunately, their scheme is shown to be insecure by Wang, Bao and Zhou [17], and subsequently, Wang et al. proposed a modified scheme that is proven secure under this stronger notion.

Yuen, Wong, Susilo and Huang [18] constructed a concurrent signature variant that supports negotiation between the initial signer and the matching signer on who will control the final binding of the ambiguous signatures. They showed that their model is compatible with the original definition by Chen et al. [6].

Very recently, Tan, Huang and Wong [15] presented the first concurrent signature scheme that is based on the standard assumption. Their ambiguity model is very similar to the one proposed by Yuen et al. [18].

1.4 Roadmap

In the next section, we review the notion of concurrent signatures due to Chen et al. and the notations used in the rest of this paper. In Section 3, we present our classification of advantage levels to the initial signer in detail, discuss their implications and present generic techniques which allows the initial signer to enjoy these advantages. In Section 4, we show how an initial signer can enjoy level 1 advantage in the asymmetric concurrent signature scheme due to Nguyen [13]. Finally, we conclude the paper in Section 5.

2 Notions and Definitions of Concurrent Signature

2.1 Notations

For a finite set $\mathcal{S}$, we will denote by $x \in_R \mathcal{S}$ the operation of selecting an element x uniformly at random from $\mathcal{S}$. If p is a positive integer, we use $\mathbb{Z}_p$ to denote the set $\{0, \ldots, p-1\}$.

2.2 Concurrent Signatures

In the following, we review the definition of concurrent signatures from [6]. A concurrent signature comprises four algorithms (SETUP, ASIGN, $\mathsf{AVERIFY}$, VERIFY). Their formal definitions are given below.

SETUP. On input security parameter 1^λ, this probabilistic algorithm outputs the description of the set of participants $\mathcal{U}$, the message space $\mathcal{M}$, the signature space $\mathcal{S}$, the keystone space $\mathcal{K}$, the keystone fix space $\mathcal{F}$, and a function $\mathsf{KGEN} : \mathcal{K} \to \mathcal{F}$. It is also assumed the public keys $\{X_i\}$ and their respective secret keys $\{x_i\}$ are also generated by this algorithm. We use π to denote additional system parameters. We assume $(\pi, \mathcal{U}, \mathcal{M}, \mathcal{S}, \mathcal{K}, \mathcal{F}, \mathsf{KGEN}, \{X_i\})$ are available to all participants while each user retain his/her own secret key x_i.

ASIGN. On input (X_i, X_j, x_i, h_2, M) where $X_i, X_j \in \{X_i\}$ such that $X_i \neq X_j$, x_i being the secret key corresponds to the public key X_i, $h_2 \in \mathcal{F}$, $M \in \mathcal{M}$, this algorithm outputs an ambiguous signature $\sigma = (s, h_1, h_2)$ on message M, where $s \in \mathcal{S}$, $h_1, h_2 \in \mathcal{F}$.

$\mathsf{AVERIFY}$. On input (σ, X_i, X_j, M), where $\sigma = (s, h_1, h_2)$, $s \in \mathcal{S}$, $h_1, h_2 \in \mathcal{F}$, X_i, X_j are distinct public keys and $M \in \mathcal{M}$, outputs 0/1. It is required that $\mathsf{AVERIFY}((s, h_1, h_2), X_i, X_j, M) = \mathsf{AVERIFY}((s, h_2, h_1), X_j, X_i, M)$.

VERIFY. On input $(k, (\sigma, X_i, X_j, M))$ such that $k \in \mathcal{K}$ and $\sigma = (s, h_1, h_2)$, it outputs 0 if $h_2 \neq \mathsf{KGEN}(k)$. Otherwise, it outputs $\mathsf{AVERIFY}(\sigma, X_i, X_j, M)$.

Below is a recap of the interactive protocol in which the above algorithms are used in the exchange of signatures in a concurrent manner amongst two participants.

1. We assume SETUP has been executed and all participants have their own key pair already. Below we describe how Alice with key pair (X_A, x_A) exchanges signatures with Bob with key pair (X_B, x_B).
2. Alice picks a random $k \in \mathcal{K}$, computes $f = \mathsf{KGEN}(k)$ and obtains $\sigma_A := (s_A, h_A, f)$ from $\mathsf{ASIGN}(X_A, X_B, x_A, f, M_A)$. Alice sends σ_A, M_A to Bob.
3. Bob verifies Alice's ambiguous signature by invoking $\mathsf{AVERIFY}(\sigma_A, X_A, X_B, M_A)$. If σ_A is valid, Bob obtains $\sigma_B := (s_B, h_B, f)$ from $\mathsf{ASIGN}(X_B, X_A, x_B, f, M_B)$. Bob sends (σ_B, M_B) to Alice.
4. Alice verifies Bob's ambiguous signature by invoking $\mathsf{AVERIFY}(\sigma_B, X_B, X_A, M_B)$. If σ_B is valid, Alice releases the keystone k. Parse S_A as $(k, \sigma_A, X_A, X_B, M_A)$ and S_B as $(k, \sigma_B, X_B, X_A, M_B)$.
5. Everybody can now verify both signatures S_A and S_B using VERIFY.

The correctness is defined in the usual manner. Specifically, if $\sigma = \mathsf{ASIGN}(X_i, X_j, x_i, f, M)$ and $S = (k, \sigma, X_i, X_j, M)$, then $\mathsf{AVERIFY}(\sigma, X_i, X_j, M) = 1$. In addition, if $f = \mathsf{KGEN}(k)$ for some $k \in \mathcal{K}$, then $\mathsf{VERIFY}(S) = 1$.

2.3 Security Model

As discussed in [6], a concurrent signature should satisfy three security requirements, namely, *unforgeability*, *ambiguity* and *fairness*. For completeness, we review these security requirements as follows.

Unforgeability The following game is used to capture the existential unforgeability of a concurrent signature.

Definition 1 (Unforgeability). *A concurrent signature is unforgeable if no PPT adversary $\mathcal{A}$ can win the following game with a challenger $\mathcal{C}$.*

Setup. $\mathcal{C}$ invokes $\mathsf{SETUP}(1^\lambda)$ for a security parameter 1^λ and obtains a set of parameters $(\pi, \mathcal{U}, \mathcal{M}, \mathcal{S}, \mathcal{K}, \mathcal{F}, \mathsf{KGEN}, \{X_i\})$ and the corresponding set of secret keys $\{x_i\}$. $\mathcal{C}$ gives the set of parameters to $\mathcal{A}$ while retaining the set of secret keys $\{x_i\}$.

Queries. $\mathcal{A}$ is allowed to issue to following queries in an adaptive manner.

1. *KGEN: $\mathcal{C}$ randomly generates $k \in_R \mathcal{K}$ and returns $f = \mathsf{KGEN}(k)$ to $\mathcal{A}$.*
2. *Keystone Reveal: On input f such that f is an output of the KGEN query, $\mathcal{C}$ returns k such that $f = \mathsf{KGEN}(k)$. Otherwise $\mathcal{C}$ returns $\perp$.*
3. *ASIGN: On input (X_i, X_j, h_2, M) such that $X_i, X_j \in \{X_i\}$, $h_2 \in \mathcal{F}$, $M \in \mathcal{M}$, $\mathcal{C}$ replies with $\mathsf{ASIGN}(X_i, X_j, x_i, h_2, M)$.*
4. *Secret Key Reveal: On input $X_i \in \{X_i\}$, $\mathcal{C}$ returns x_i.*

Output. Finally $\mathcal{A}$ outputs a signature $\sigma^* = (s^*, h^*, f^*)$, a message M^* and two public keys X_c^*, X_d^*. $\mathcal{A}$ wins the game if $\mathsf{AVERIFY}(\sigma^*, X_c^*, X_d^*, M^* = 1)$ and either one of the following is true:

- *(X_c^*, X_d^*, f^*, M^*) is not an input to the ASIGN query and X_c^*, X_d^* is not an input to the secret key reveal query.*
- *$\mathcal{A}$ has not made any ASIGN query of the form (X_c^*, X, f^*, M^*) for all $X \in \{X_i\} \setminus \{X_c^*\}$, no secret key reveal query was made with input X_c^* and f^* is the output of KGEN query or $\mathcal{A}$ also outputs k^* such that $f^* = \mathsf{KGEN}(k^*)$.*

Ambiguity The following game is used to capture ambiguity of a concurrent signature.

Definition 2 (Ambiguity). *A concurrent signature is ambiguous if no PPT adversary $\mathcal{A}$ can win the following game with a challenger $\mathcal{C}$.*

Setup. Same as Setup in the game in Definition 1.

Phase 1. $\mathcal{A}$ is allowed to made a sequence of KGEN, Keystone Reveal, ASIGN and Secret Key Reveal query, which are answered as in the game of subsection 2.3.

Challenge. $\mathcal{A}$ outputs two public keys X_i, X_j and a message M as challenge. $\mathcal{C}$ randomly picks $k \in_R \mathcal{K}$ and computes $f = \mathsf{KGEN}(k)$. $\mathcal{C}$ then flips a fair coin $b \in_R \{0, 1\}$. If $b = 0$, $\mathcal{C}$ computes $\sigma_0 = \mathsf{ASIGN}(X_i, X_j, x_i, f, M)$. Otherwise, $\mathcal{C}$ computes $(s, h, f) = \mathsf{ASIGN}(X_j, X_i, x_j, f, M)$ and parse σ_1 as (s, f, h). $\mathcal{C}$ returns σ_b to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ can make another sequence of queries as in phase 1.

Output. Finally $\mathcal{A}$ outputs a guess bit b'. $\mathcal{A}$ wins the game if $b = b'$ and $\mathcal{A}$ did not make any Keystone Reveal query on input f or h.

Fairness The following game is used to capture fairness of a concurrent signature.

Definition 3 (Fairness). *A concurrent signature is fair if no PPT adversary $\mathcal{A}$ can win the following game with a challenger $\mathcal{C}$.*

Setup. Same as Setup in the game of subsection 2.3

Queries. $\mathcal{A}$ is allowed to made a sequence of KGEN, Keystone Reveal, ASIGN and Secret Key Reveal query, which are answered as in the game of subsection 2.3.

Challenge. $\mathcal{A}$ outputs two public keys X_i, X_j and two messages M_i, M_j, together with $\sigma_i = (s_i, h_1, h_2)$ such that $\mathsf{AVERIFY}((s_i,\ h_1,\ h_2),\ X_i,\ X_j,\ M_i) = 1$. $\mathcal{C}$ returns $\sigma_j = (s_j, h_3, h_2) = \mathsf{ASIGN}(X_j, X_i, x_j, h_2, M_j)$.

Output. Finally $\mathcal{A}$ either outputs a value k. $\mathcal{A}$ wins the game if $f = \mathsf{KGEN}(k)$ such that f was a previous output from KGEN query and no Keystone Reveal query on f was made.

3 Abusing Fairness in Concurrent Signatures

In this section, we discuss the various advantage levels enjoyed by the initial signer, their implications and how they could be acquired. At the high level, it is often the case that exhibiting such advantage requires the use of zero-knowledge proof [9], which is reviewed as follows.

3.1 Zero-Knowledge Proof

A zero-knowledge proof [9] is an interactive protocol for one party, the prover, to prove to another party, the verifier, that some statement is true, without revealing anything other than the veracity of the statement. In [8], it has been shown that, assuming the existence of one-way function, one can create a zero-knowledge proof system for the NP-complete graph coloring problem with three colors. Since every problem in NP can be efficiently reduced to this problem, it means that all problems in NP have zero-knowledge proofs. Later in [5], it has been shown that anything that can be proved by an interactive proof system can be proved with zero knowledge.

3.2 Advantage Level 0

Level 0 advantage is inherent in concurrent signatures, as the initial signer, Alice, is in possession of the keystone, which is under the full control of Alice on when and whether the keystone will be released to the public. Thus, the primitive is not suitable if the matching signer will be at a disadvantage if withholding the keystone or delaying the release of the keystone would cause harm to the matching signer.

The Implications. Consider the tendering systems as discussed in the seminal paper of Chen et al. [6]. They described a scenario where A has a bridge-building contract that she would like to put out to tender. Suppose there are two companies B and C that wish to put in proposals to win this contract. In this scenario, B acts as the initial signer, as this is to prohibit A to show this contract to C to get a better proposal from C. We note that in this particular scenario, B has the full control of the keystone, since the keystone is selected by B. Therefore, if at the end of the tender, if A would like to select B as the winner of the tender, B may still have the liberty for not completing the contract by not releasing the keystone, and hence, it is unfair to A.

3.3 Advantage Level 1

The Abuse. Assume Alice is a malicious initial signer, whose purpose is to convince a third party Carol that the matching signer Bob and herself are about to exchange signatures on messages M_A and M_B. Assume Alice and Bob have completed step 3 of the concurrent signature protocol (as described in Section 2.2). That is, Alice is in possession of a keystone k and the ambiguous signature from Bob $\sigma_B := (s_B, h_B, f)$ on message M_B. At the same time, Bob is in possession of Alice's ambiguous signature $\sigma_A := (s_A, h_A, f)$ on message M_A. In our generic attack, Alice reveals σ_A, σ_B, M_A, M_B to Carol and then conducts a zero-knowledge proof-of-knowledge of the value k with Carol such that

$$f = \mathsf{KGEN}(k).$$

Carol is convinced that Alice and Bob are exchanging signatures on messages M_A and M_B and that Alice has the ability to complete the transaction.

The Implications. Consider an open auction [17, 18] in which Alice's ambiguous signature is a contract to sell a certain goods while Bob's ambiguous signature is a contract of a bid. Level-1 advantage allows Alice to convince Carol that she has the ability to seal the contract with Bob and the bid is specified in M_B. This allows Alice to safely urge Carol for a higher bid, and Bob is at a disadvantage.

We note that this implication is also applicable to the tendering systems as discussed previously. However, in this scenario, let us consider the case where A, who would like to put the bridge-building tender, is the initial signer. Hence, the company B and C will act as the matching signers. In this setting, A will take the advantage level 1 to convince C about B's tender, so that C will increase the value of her tender, and hence, disadvantaging B.

3.4 Advantage Level 2

The Abuse. Assume Alice is a malicious initial signer, whose purpose is to convince a third party Carol that the matching signer Bob has committed to message M_B. Assume Alice and Bob have completed step 3 of the concurrent signature protocol (as described in Section 2.2). That is, Alice is in possession of a keystone k and the ambiguous signature from Bob $\sigma_B := (s_B, h_B, f)$ on message M_B. At the same time, Bob is in possession of Alice's ambiguous signature $\sigma_A := (s_A, h_A, f)$ on message M_A. Our observation on the incompleteness of the original fairness definition in [6] arises from the fact that to convince Carol about Bob's commitment to M_B *does not necessarily* involve outputting some maliciously crafted keystone $\hat{k}$. Specifically, in our generic attack, Alice conducts a zero-knowledge proof-of-knowledge of the tuple (k, σ_B) with Carol such that the following statements are true:

1. $f = \mathsf{KGEN}(k)$, and
2. $\mathsf{AVERIFY}(\sigma_B, X_A, X_B, M_B) = 1$.

This would be sufficient to convince a third party about *Bob's intention* to sign a message M_B, without revealing anything about Alice's intention, thus undermining the fairness guarantees of the concurrent signature schemes, and put Bob into disadvantage.

The Implications. Note that in the zero-knowledge proof, Alice does not reveal the keystone k, the keystone fix f, nor Bob's ambiguous signature. Thus, even if Carol is presented with the secret key of Bob, the ambiguous signatures σ_A and σ_B from Bob, she could not conclude that Alice has committed to M_A. In other words, the only thing that Carol can be assured of is that Alice is in possession of a signature from Bob on message M_B. Bob is left at an unfair position.

We would like to remark that both level 1 and 2 advantages are outside the security definition presented in Section 2.3. We also do not claim that existing concurrent signature schemes are broken for two reasons. Firstly, they are not within the security model. Secondly, even though they are theoretically possible, it is not always feasible. Thus, people should bear in mind any concurrent signature following this syntax *could* be abused by the initial signer, and hence, the adoption of concurrent signatures in application scenario should be examined to make sure that either the level 1 and 2 advantages of the initial signer are acceptable or that the advantages cannot be claimed efficiently.

4 Abusing Fairness in Asymmetric Concurrent Signatures

In this section, we demonstrate a practical abuse in advantage level 1 of the asymmetric concurrent signature due to Nguyen [13]. In this abuse, the initial signer can convince any verifier that he/she has the ability to make both ambiguous signatures verifiable. We would like to stress again that the attack is *outside* their original model and we therefore do not claim that we break Nguyen's original scheme.

4.1 Review of Nguyen's asymmetric concurrent signatures

For completeness, we will first review Nguyen's scheme.

Setup. Let $\mathbb{G} = \langle g \rangle$ be a group of prime order p. The key pair of Alice and Bob are respectively $(X_A = g^{x_A}, x_A) \in \mathbb{G} \times \mathbb{Z}_p$, $(X_B = g^{x_B}, x_B) \in \mathbb{G} \times \mathbb{Z}_p$. Let $H : \mathbb{G} \times \{0,1\}^* \to \mathbb{Z}_p$ be a hash function that would be modeled as random oracle.

Generation of Alice's ambiguous signature. On input a message $M_A \in \{0,1\}^*$ and Bob's public key X_B, Alice randomly generates $r \in_R \mathbb{Z}_p$ and computes $c = H(g^r, M_A)$, $s = g^{r+cx_A}$. Alice sets her ambiguous signature as (c, s) and sends it to Bob. The keystone is defined as $k = r + cx_A$ such that $s = g^k$.

Generation of Bob's ambiguous signature. On input a message $M_B \in \{0,1\}^*$ and an ambiguous signature (c, s) on message M_A from Alice, Bob first check if $c = H(sX_A^{-c}, M_A)$. Bob continues if the check is successful. He computes $s' = s^{x_B}$, $c' = H(s'g^{r'}, M_B)$ and $k' = \frac{r'-c'}{x_B}$. He sets his ambiguous signature as (c', s', k') and sends it to Alice.

Binding of both signatures. Upon receiving (c', s', k') from Bob, Alice first checks if

$$c' = H(g^{c'} X_B^{k'} s', M_B)$$

and $s' = X_B^k$. If the check is successful and if Alice decides to have both signatures binded, she releases the value k. Both signatures are now publicly verifiable by checking the following verification equations.

$$\begin{gathered}\text{Verification of Alice's signature:}\\ c \overset{?}{=} H(X_A^{-c} s, M_A) \ \wedge \ s \overset{?}{=} g^k\\ \text{Verification of Bob's signature:}\\ c' \overset{?}{=} H(g^{c'} X_B^{k'} s', M_B) \ \wedge \ s' \overset{?}{=} X_B^k\end{gathered}$$

Remark. Nguyen's construction is asymmetric in the sense that Alice's signature and Bob's signature are of different form with different verification equations.

4.2 A Concrete Level-1 Abuse on Nguyen's Scheme

Given the pair of ambiguous signatures (c, s), (c', s', k') on message M_A and M_B respectively, Alice, who is in possession of the keystone k satisfying the relationship $s = g^k$ and $s' = X_B^k$ can convince a third party Carol that she has the ability to bind both ambiguous signatures by proving she knows the value k in zero-knowledge. The full proof protocol is shown below.

1. Alice sends both (c, s), (c', s', k') M_A, M_B to Carol.
2. Carol checks $c = H(X_A^{-c} s, M_A)$ and $c' = H(g^{c'} X_B^{k'} s', M_B)$. If yes, she randomly generates two values $t_1, t_2 \in_R \mathbb{Z}_p$ and sends $T_0 = g^{t_1} h^{t_2}$ to Alice.
3. Alice randomly generates a value $t \in_R \mathbb{Z}_p$, computes $T_1 = g^t$, $T_2 = X_B^t$ and sends T_1, T_2 to Carol.
4. Carol sends t_1, t_2 to Alice
5. Alice checks if $T_0 = g^{t_1} h^{t_2}$. If yes, she computes $z = t - t_1 k$ and sends z to Carol.
6. Carol checks if $T_1 = s^{t_1} g^z$ and $T_2 = s'^{t_1} X_B^z$ and accepts the the proof if both equations hold.

5 Conclusion

We pointed out the advantage gained by the initial signer in a concurrent signature scheme, and classified into three levels. In fact, any concurrent signature satisfying Chen, Kudla and Paterson's syntax can be abused in different ways. This is a very important observation in particular where concurrent signatures are used in different scenarios. Cautions must be exercised when concurrent signatures are to be adopted in real applications to ensure either the matching signer can accept the inherent unfairness of concurrent signatures or it is hard for the initial signer to claim the advantage. In particular, we demonstrated that concurrent signatures are not suitable for tendering systems (in contrast to the seminal paper of Chen et al. [6]).

Acknowledgements

We would like to thank anonymous reviewers of ACISP 2013 who have helped us to improve the clarity and quality of this paper.

References

1. M. Abe, M. Ohkubo, and K. Suzuki. 1-out-of-n Signatures from a Variety of Keys. *Advances in Cryptology - Asiacrypt 2002, Lecture Notes in Computer Science 2501*, pages 415 – 432, 2002.
2. N. Asokan, M. Schunter, and M. Waidner. Optimistic protocols for fair exchange. In *Proc. 4th ACM Conference on Computer and Communications Security*, pages 8–17, 1997.
3. N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. *Advances in Cryptology-Eurocrypt 1998, Lecture Notes in Computer Science 1403*, pages 591 – 606, 1998.
4. N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. *IEEE Journal of Selected Areas Communications, vol. 18, no. 4*, pages 593 – 610, Apr. 2000.
5. M. Ben-Or, O. Goldreich, S. Goldwasser, J. Håstad, J. Kilian, S. Micali, and P. Rogaway. Everything provable is provable in zero-knowledge. In *Proceedings of the 8th Annual International Cryptology Conference on Advances in Cryptology*, pages 37–56, London, UK, 1990. Springer-Verlag.
6. L. Chen, C. Kudla, and K. G. Paterson. Concurrent signatures. *Advances in Cryptology - Eurocrypt 2004, Lecture Notes in Computer Science 3027*, pages 278 – 305, 2004.
7. Y. Dodis, P. J. Lee, and D. H. Yum. Optimistic fair exchange in a multi-user setting. In T. Okamoto and X. Wang, editors, *Proceedings of Public Key Cryptography 2007*, volume 4450 of *Lecture Notes in Computer Science*, pages 118–133. Springer, 2007. Also at Cryptology ePrint Archive, Report 2007/182.
8. O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *J. ACM*, 38:690–728, July 1991.
9. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.
10. Q. Huang, D. S. Wong, and W. Susilo. Efficient designated confirmer signature and DCS-based ambiguous optimistic fair exchange. *IEEE Transactions on Information Forensics and Security*, 6(4):1233–1247, 2011.
11. Q. Huang, D. S. Wong, and W. Susilo. The construction of ambiguous optimistic fair exchange from designated confirmer signature without random oracles. In *Proceedings of Public Key Cryptography 2012*, volume 7293 of *Lecture Notes in Computer Science*, pages 120–137. Springer, 2012.
12. Q. Huang, G. Yang, D. S. Wong, and W. Susilo. Ambiguous optimistic fair exchange. In *Advances in Cryptology - ASIACRYPT 2008*, volume 5350 of *Lecture Notes in Computer Science*, pages 74–89. Springer, 2008.
13. K. Nguyen. Asymmetric concurrent signatures. *Information and Communications Security (ICICS 2005), Lecture Notes in Computer Science 3783*, pages 181 – 193, 2005.
14. W. Susilo, Y. Mu, and F. Zhang. Perfect concurrent signature schemes. *Information and Communication Security (ICICS 2004), LNCS 3269*, pages 14 – 26, 2004.
15. X. Tan, Q. Huang, and D. S. Wong. Concurrent signature without random oracles. Cryptology ePrint Archive, Report 2012/576, 2012. `http://eprint.iacr.org/`.
16. D. Tonien, W. Susilo, and R. Safavi-Naini. Multi-party concurrent signatures. *The 9th Information Security Conference (ISC 2006), LNCS 4176*, 2005.
17. G. Wang, F. Bao, and J. Zhou. The fairness of perfect concurrent signature. *Information and Communication Security(ICICS 2006), LNCS 4307*, pages 435 – 451, 2006.
18. T. H. Yuen, D. S. Wong, W. Susilo, and Q. Huang. Concurrent signatures with fully negotiable binding control. In X. Boyen and X. Chen, editors, *ProvSec*, volume 6980 of *Lecture Notes in Computer Science*, pages 170–187. Springer, 2011.