# Multi-Party Concurrent Signature Scheme Based on Designated Verifiers

Jianni Xushuai, Zhihong Zhou
Shanghai Jiao Tong University/ Shanghai Key Laboratory of Integrate Administration Technologies for Information Security, School of Information Security Engineering, Shanghai, China
Email: zhouzhihong@sjtu.edu.cn (ZH Zhou)

Wen Qin[1], Qiongxi Jiang[1], Nanrun Zhou[1, 2]
1. Nanchang University/Department of Electronic Information Engineering, Nanchang, China
2. National Engineering Laboratory for Disaster Backup and Recovery, Beijing University of Posts and Telecommunications, Beijing 100876,China
Email: nrzhou@ncu.edu.cn (NR Zhou)

***Abstract*—Fair exchange signature is of importance in the field of the open-network-based security applications. A new multi-party *concurrent signature* (MPCS) scheme based on *designated verifiers* is introduced, which features *fairness* and unforgeability based on the hardness of the Computational Diffie-Hellman (CDH) assumption in the random oracle model. In this scheme, each signer has the right to choose randomly his/her own individual keystone and retrieve all other individual keystones by the Extraction algorithm. If all signers release their own individual keystones, all signatures can be bound. There is not a decisive signer or a more power signer in selecting and releasing keystones. Therefore, the situation of keystones switched by dishonest signers can be effectively avoided and the fairness of the MPCS scheme is also apparently improved. Our MPCS scheme is proved to be secure and can counteract the adaptive chosen message attack.**

***Index Terms*—Designated verifiers signature, Fairness, Concurrent signature, Information security**

## I. INTRODUCTION

With the rapid growth of electronic commerce [1] and electronic governance [2] affair nowadays, fair exchange turns out to be an increasingly important topic, such as e-commerce payment protocol, electronic contract signing protocol and certified e-mail delivery. In recent years, many fair exchange signature schemes (FESS) have been reported to make two individuals [3] or even two parties from different groups [4] exchange their signatures fairly. A third party is involved to judge a dispute in the FESS. Nevertheless, it is also an increasingly painful problem for three or more participants to exchange fairly their signatures or goods on a contract, since these participants may be dishonest to implement the contract.

There are mainly two different kinds of solutions to the problem of fair exchange signatures. One assumes that both parties have equivalent computation resources, and inefficient exchange signature "bit-by-bit" for many interactive rounds [5-7]. Nevertheless, these methods can't provide complete fairness, since one signer often has an advantage of one more bit than the other signer at the end of protocol. Therefore, these methods may be too interactive to apply widely. The other involves a TP, a TTP or an arbitrator to run or handle disputes between signers if necessary [8-11]. However, these schemes may cause the bottleneck problem, since they require a dispute-resolving TTP, whose function is beyond the demands of a normal Certification Authority. L. Q. Chen et al introduced concurrent signatures to solve the problem of fair exchange signatures in 2004 [12]. The concurrent signature scheme having the ambiguity property of the ring signatures [13, 14] is a weaker version of two-party fair exchange without any third trusted party and many interactive exchange rounds. Such signature schemes allow two entities to produce and exchange two signatures, which are ambiguous until an extra piece of information, i.e., the keystone, is released by one of the parties. After the keystone is publicly known, the signer for each signature is identified and both signatures can instantly bind to their corresponding signers. W. Susilo et al pointed out that any third party can differentiate the real signer of a signature before the keystone is released when the initial signer Alice and the matching signer Bob are known to be honest players [15] and they extended concurrent signature to a stronger notion of perfect concurrent signature, which guarantees full ambiguity of the concurrent signature even if two entities are known to be trustworthy. G. L. Wang et al pointed out that there is an attack in the perfect concurrent signature schemes [16]. If the keystone releasing to Bob is prepared carefully by Alice, then Bob's signature can be bound while Alice's signature cannot. W. Susilo et al proposed a tripartite concurrent signature based on bilinear pairings [17]. However, it seems too complex to extend to the multi-party case. S. S. M. Chow et al proposed two ID-based perfect concurrent signature schemes based on two major paradigms of ID-based ring signature schemes [18]. Z. J. Huang et al pointed out that the two identity-based perfect concurrent signature schemes are unfair, since the initial signer could cheat the matching one [19]. K. Nguyen proposed


doi:10.4304/jcp.8.11.2823-2830

asymmetric concurrent signature schemes based on Schnorr signature scheme and Schnorr-like signature scheme [20]. In 2010, J. W. Liu et al [21] proposed FESS, which have higher efficiency than concurrent signature schemes. However, FESS can not overcome the weakness of concurrent signature, i.e., the keystone is still controlled by the initial signer.

D. Tonien et al proposed a multi-party concurrent signature (MPCS) scheme with ring signatures and bilinear pairings in 2006 [22]. C. Shieh et al pointed out that the MPCS scheme proposed by Tonien et al did not achieve the goal of a concurrent signature [23], and proposed a fair MPCS scheme providing a fairer property and flexibility in a transaction. Using bilinear parings, L. L. Wang constructed a MPCS scheme [24], where the short ring signature is constructed based on dynamic accumulators. An MPCS scheme was presented by X. Tan et al [25], where the concurrent signatures can be converted to ordinary signatures containing not any information of keystone to remain unlinkable when the keystone was released. L. Q. Chen et al noted that it will move closer to the full solution to the multi-party fair exchange problem if the MPCS scheme can be constructed and modeled correctly [12]. In fact, the MPCS scheme can also be constructed if we make use of the main technique called multi-designated verifiers [26]. Combining identity-based cryptographic system with the designated verifier, B. Y. Kang et al proposed an identity-based designated verifier signature scheme [27], where the designated verifier signature can not be verified by any third party. Moreover, the designated verifier proofs are also proposed in [28]. The designated verifier signatures are only convinced by the intended recipient, who has a solely legal and designated public-key. However, D. Tonien et al noted that the construction from designated verifier proofs is not straightforward and it cannot achieve the required properties of concurrent signatures. Actually, F. Laguilaumie [26] noted that their scheme satisfies the properties of correctness, existential unforgeability against a chosen message attack, privacy of the signer's identity and source hiding. Moreover, it is not necessary for the secret information to be additionally encrypted and then be shared between the participants. Therefore, the participants have the same right to learn the secret information to ensure the fairness between the participants. Then a new MPCS scheme can be constructed effectively based on the above properties.

To ensure the fairness of MPCS scheme, a new model with keystones adopted in our two-party concurrent signature scheme [29] can be extended to the multi-user scenario. In the previous models, the initial signer has a decisive power or more power over the other users to select a single keystone. Motivated by this, we propose an MPCS scheme based on designated verifiers in this paper, in which all signers can select their own individual keystones to achieve the fairness. The binding of the signatures happens after one of the signers releases all their individual keystones. In the actual scenario, our MPCS scheme can be carried out over open networks and all signers do not need mutual trust. Once an initial signer implements this MPCS scheme, all signers have the same right to select themselves individual keystones and release all individual keystones to achieve the fairness of participation on a contract.

The rest of this paper is organized as follows. The related basic concepts are described in section 2. The proposed multi-party concurrent signature scheme based on designated verifiers is introduced in section 3. The performances of our proposed scheme are analyzed in section 4. And a brief conclusion is reached in section 5.

## Ⅱ. BASIC CONCEPTS

### A. Bilinear Pairings

Let $G_1$ and $G_2$ be a cyclic additive group and a cyclic multiplicative group, respectively. Let $e: G_1 \times G_1 \to G_2$ be a bilinear mapping with the following properties [30]:

(1) Bilinearity: for any $\alpha, \beta \in Z_q$ and $P, Q \in G_1$, $e(\alpha P, \beta Q) = e(P, Q)^{\alpha\beta}$;

(2) Non-degeneracy: there exists $P, Q \in G_1$, such that $e(P, Q) \neq 1$;

(3) Computability: there exists an efficient algorithm to calculate $e(P, Q)$ for all $P, Q \in G_1$.

A bilinear pairing instance generator is defined as a probabilistic polynomial time algorithm $\Lambda$ that takes a security parameter $l$ as input and returns a uniformly random tuple parameter $(q, P, G_1, G_2, e)$ of bilinear parameters, including a prime number $q$ of size $l$, a generator $P$ of $G_1$.

### B. CBDH Assumption

Given $(P, xP, yP, zP)$ for some $x, y, z \in Z_q^*$, compute $g = e(P, P)^{xyz} \in G_2$. The CBDH assumption over $G_2(G_1, G_2, e)$ states that any poly-time algorithm $A$ has a negligible success probability in solving the CBDH problem, that is to say, $\Pr[A(P, xP, yP, zP) = e(P, P)^{xyz}]$ is negligible in $\log q$, where the probability is over the random choice of $x, y, z \in Z_q^*$ and $P \in G_1$ [29].

### C. CDH Assumption

Given $(P, xP, yP)$ for some $x, y \in Z_q^*$, compute $xyP \in G_1$. The CDH assumption over $G_1$ states that any poly-time algorithm $A$ has a negligible success probability in solving the CDH problem, that is to say, $\Pr[A(P, xP, yP) = xyP]$ is negligible in $\log q$, where the probability is over the random choice of

$x, y \in Z_q^*$ and $P \in G_1$. The CBDH assumption is much stronger than the CDH assumption since one can calculate $xyP$ and further calculate $e(P,P)^{xyz} = e(zP, xyP)$ with $zP$ for a given $(P, xP, yP)$ [30].

## III. Multi-party Concurrent Signature Scheme Based on Designated Verifiers

Our construction is inspired by the jobs in [17, 26]. In our MPCS scheme, $n$ participants are bound with the security parameter $l$. The outputs of the procedures such as Setup, Keystone Generation and ASign are probabilistic. However, the procedures including Extraction, AVerify and BindingVerify have the deterministic outputs. Our MPCS scheme works as follows:

(1) Setup: On inputting a security parameter $l$, descriptions of the set of the participants $U$, the message space $M$, the signature space $S$, the keystone space $K$, the encrypted keystone $F$, and a hash function $H: \{0,1\}^* \times G_1 \times G_1 \to G_1$ are outputted. The initial algorithm $\Lambda$ selects a uniformly random tuple parameter $(q, G_1, G_2, e, P)$ of bilinear parameters, including a prime number $q$ of size $l$, a cyclic additive group $G_1$ of order $q$, a multiplicative group $G_2$ of order $q$, a bilinear map $e: G_1 \times G_1 \to G_2$ and a generator $P$ of $G_1$. The Setup algorithm sets $K = G_1$, $F = G_1$, $M = \{0,1\}^*$. A function KGEN is defined as $K \to F$.

The Setup algorithm also outputs the public parameters, together with all public keys $\{K_{\mathrm{P}i}\}$ of the participants, where each participant retains his/her private key. Each participant $U_j$, for $j \in \{1, \cdots, n\}$, randomly selects a number $k_{\mathrm{S}j} \in Z_q^*$ as his/her secret key, then the corresponding public key is $K_{\mathrm{P}j} = k_{\mathrm{S}j} P$. The public key tuple is $\{K_{\mathrm{P}i} \mid i = 1, \cdots, n\}$.

(2) Keystone Generation: One of the participants, as an initial signer $j$, selects a secret number $r_j \in Z_q^*$ associated with the signer $j$ at random. His/Her keystone is $k_j = r_j P$. Similarly, the individual keystone of the other participants is $k_i = r_i P$, for $i \neq j$. The encrypted individual keystones are produced by the public key of the verifiers, i.e., $C_j^i = r_j K_{\mathrm{P}i}$ for $i = 1, \cdots, n$, $i \neq j$. In addition, the signers in the scheme have the same right to choose their own individual keystones and produce the corresponding encrypted individual keystones.

(3) ASign: Given $m_j = \{0,1\}^*$, the initial signer $j$ computes the sum of partial public key $K_{\mathrm{P}j}^{'}$.

$$K_{\mathrm{P}j}^{'} = \sum_{i=1}^{n} K_{\mathrm{P}i} - K_{\mathrm{P}j},\ j = 1, \cdots, n. \qquad (1)$$

The signer gets $C_j^j = H(m_j \,||\, K_{\mathrm{P}j}^{'} \,||\, k_j)$. The ASign algorithm also selects an $r_j^{'} \in Z_q^*$ at random and computes $Q_j^1 = k_{\mathrm{S}j}^{-1}(C_j^j - r_j^{'} K_{\mathrm{P}j}^{'})$ and $Q_j^2 = r_j^{'} P$. The signature $\sigma_j$ of $m_j$ is $(Q_j^1, Q_j^2, C_j^1, \cdots, C_j^n)$. Hence, the signer sends the signature to the only designated verifiers (the other participants).

(4) Extraction: Given $m_j = \{0,1\}^*$ and $\sigma_j$, the other participants (the designated verifiers) can take advantage of $C_j^i k_{\mathrm{S}i}^{-1}$ to retrieve $k_j = r_j P$. The Extraction algorithm is crucial for the other participants to ensure the truth of the individual keystone released from the signer $j$ in our MPCS scheme. In other words, the designated verifiers can obtain the individual keystone, while the outsiders cannot.

(5) AVerify: After the other participants compute the individual keystone of $j$, if the other participants verify $e(K_{\mathrm{P}i}, k_j) = e(C_j^i, P)$ is true, i.e., the first part of the AVerify algorithm outputs "accept", then the other participants compute $C_j^j = H(m_j \,||\, K_{\mathrm{P}j}^{'} \,||\, k_j)$ and check whether $e(C_j^j, P)$ is equal to $e(Q_j^1, K_{\mathrm{P}j}) e(Q_j^2, K_{\mathrm{P}j}^{'})$ or not. If yes, the second part of the AVerify algorithm outputs "accept", then the other participants generate $\sigma_i$ of $m_i$ for $i \in \{1, \cdots, n\}$, $i \neq j$ by the above ASign algorithm. Once all signatures generated by all participants are verified successfully by the Averify algorithm, one of the participants will release all individual keystones to accomplish binding. The AVerify algorithm can be used to check the validity of an anonymous signature by an insider.

(6) BindingVerify: The algorithm accepts input $(k_1, \sigma_1, m_1, k_2, \sigma_2, m_2, \cdots, k_n, \sigma_n, m_n)$, consequently checks the following two conditions and outputs reject if any condition fails, i.e., for each $j \in \{1, 2, \cdots, n\}$,

$C_j^j = H(m_j \,||\, K_{\mathrm{P}j}^{'} \,||\, k_j)$;

$e(C_j^j, P) = e(Q_j^1, K_{\mathrm{P}j}) e(Q_j^2, K_{\mathrm{P}j}^{'})$.

The algorithm can be used by anyone to check the validity of a list of $n$ signatures and to bind the identities of $n$ signers.

## IV. Performance Analyses

*A. Correctness*

**Theorem 1** If the signatures $\{\sigma_j \mid j=1,\cdots,n\}$ generated by the proposed MPCS scheme pass the AVerify algorithm and the BindingVerify algorithm, then our MPCS scheme is correct.

**Proof.** Given $m_j=\{0,1\}^*$, the sum of partial public key $K_{\mathrm{P}j}^{'}$ and the encrypted individual keystone $C_i=r_jK_{\mathrm{P}i}$ for $i\in\{1,\cdots,n\}$ and $i\neq j$, then the signer $j$ can gain $\sigma_j=(Q_j^1,Q_j^2,C_j^1,\cdots,C_j^n)$ by the ASign algorithm. Therefore, the other participants can make full use of the knowledge of the bilinear to verify $\sigma_j$ by $e(K_{\mathrm{P}i},k_j)=e(C_j^i,P)$, the result turns out "accept" with an overwhelming probability. If so, $j$ verifies whether $e(C_j^j,P)$ is equal to $e(Q_j^1,K_{\mathrm{P}j})e(Q_j^2,K_{\mathrm{P}j}^{'})$ or not. The probability for any attackers to solve the CDH problem successfully is almost negligible under the complexity assumptions. Hence the other participants can successfully check the signature with an overwhelming probability. Further, once their own individual keystones are released by one of the signers, the outsiders can easily check the validity of keystones by $C_j^j=H(m_j\|K_{\mathrm{P}j}^{'}\|k_j)$, and the outsiders can further check the signatures of correctness by the AVerify algorithms and the BindingVerify algorithm. Then the correctness of our signature scheme is deduced as follows:

(1) $C_j^ik_{\mathrm{S}i}^{-1}=k_j=r_jP$.

(2) $C_j^j=H(m_j\|K_{\mathrm{P}j}^{'}\|k_j)$.

(3) $e(K_{\mathrm{P}i},k_j)=e(C_j^i,P)$ for $i\in\{1,\cdots,n\}$ and $i\neq j$.

(4)

$$\begin{aligned}e(C_j^j,P)&=e(Q_j^1,K_{\mathrm{P}j})e(Q_j^2,K_{\mathrm{P}j}^{'})\\&=e(k_{\mathrm{S}j}^{-1}(C_j^j-r_j^{'}K_{\mathrm{P}J}^{'}),k_{\mathrm{S}j}P)\\&\quad e(r_j^{'}P,P(k_{\mathrm{S}1}+\cdots+k_{\mathrm{S}j-1}+k_{\mathrm{S}j+1}+\cdots+k_{\mathrm{S}n}))\\&=e(C_j^j-r_j^{'}K_{\mathrm{P}J}^{'},P)\\&\quad e(r_j^{'}P(k_{\mathrm{S}1}+\cdots+k_{\mathrm{S}j-1}+k_{\mathrm{S}j+1}+\cdots+k_{\mathrm{S}n}),P)\\&=e(C_j^j-r_j^{'}K_{\mathrm{P}J}^{'}+r_j^{'}K_{\mathrm{P}J}^{'},P)\\&=e(C_j^j,P)\end{aligned}$$

*B. Unforgeability*

**Theorem 2** Let $l$ be a security parameter and $J$ be an existential unforgeability against a chosen message attack (EF-CMA) adversary, which has a probability $\mathrm{Adv}_{\text{EF-CMA}}^J$ of generating an existential forgeable signature within running time $t$ and a probability $\mathrm{Succ}_{\mathrm{CDH}}^J(l)$ of solving the CDH problem within $t^{'}$. The time to execute the existential forgery attack is defined by $t^{'}\le 2(t+q^H+2q^{\Sigma}+O(1)T_{G_1}+q^{\Sigma}T_{G_2})$. $q^H$, $q^{\Sigma}$ and $q^{\phi}$ denote the number of queries of random oracle $H$, signing oracle $\Sigma$ and verifying oracle $\phi$, respectively. $T_{G_1}$ and $T_{G_2}$ denote the time complexity to perform a scalar multiplication in $G_1$ and an exponentiation in $G_2$, respectively. Assume the hardness of the CDH problem, then our MPCS scheme is existentially unforgeable under a chosen message attack in the random oracle model.

**Proof.** We consider an EF-CMA adversary $J$ that outputs an existential forgery $(m^*,\sigma^*)$ with probability $\mathrm{Adv}_{\text{EF-CMA}}^J$ within running time $t$. Let $\gamma=xP$, $\Phi=yP$ and $\psi=xyP$ be the CDH challenge. Assume the adversary $J$ is given the following resources:

(1) Public key: The adversary is given the public tuple $\{K_{\mathrm{P}i}\mid i=1,\cdots,n\}$.

(2) Hash queries: At anytime, the adversary can ask a hash value for any input. For each hash function, we will maintain a hash list so that the hash outputs could be consistent.

(3) ASign queries: The adversary can corrupt up to $n-2$ identifiable receivers to obtain their secret keys, even if he can know an individual keystone of a participant. The adversary can check the validity of the signature by himself, but he is not allowed to query the verifying oracle $\phi$ with any signature on the message. The adversary can generate a signature of the secret key $k_{\mathrm{S}i}$ with a valid individual keystone from the signing oracle $\sum$.

The forgery game between $J$ and $L$ is as follows.

(1) Setup: Given a security parameter $l$, $L$ runs an initial algorithm $\Lambda$ to obtain the public information. Public keys of the participants and the system parameters are published. Assume a challenger $L$ is the only verifier not among the corrupted verifiers. The challenger's public key is $K_{\mathrm{P}L}$, which is replaced by $\alpha\gamma-\sum_{i\neq L}K_{\mathrm{P}i}$, $\alpha\in Z_q^*$. The adversary's public key is $K_{\mathrm{P}L}$, which is replaced by $\gamma=xP$. If the adversary can compute the point $yP$ from these points, then the CDH problem can be solved [31].

(2) Keystone Queries: $J$ requests $L$ to select an individual keystone $k_J\in K$ to generate an encrypted keystone $C_J^L=r_JK_{\mathrm{P}L}$. $J$ also selects his own

individual keystone $k_J \in K$ and computes the encrypted keystone $C_J^L$ by running the $\mathrm{KGEN}(k_J)$ function.

(3) KReveal Queries: $J$ requests $L$ to reveal the individual keystone $k_J$ to produce an encrypted keystone $C_J^L$ in a previous Keystone Query. If $C_J^L$ was not asked before, then $L$ outputs invalid, or else $k_J$.

(4) Hash Queries: $J$ queries the random oracle $H$ at any time. $L$ checks his H-list when $(m^*, K_{\mathrm{PL}}, \{K_{\mathrm{P}i}\}_{J\neq i}, C_J^L, k_J)$ are requested. If the query of $(m^*, K_{\mathrm{PL}}, \{K_{\mathrm{P}i}\}_{J\neq i}, C_J^L, k_J)$ exists, then $L$ returns the corresponding value from the list. Otherwise, $L$ selects a random $r_J^{'} \in Z_q^*$ and computes $C_J^J = r_J^{'}\psi$. The value $(m^*, K_{\mathrm{PL}}, \{K_{\mathrm{P}i}\}_{J\neq i}, C_J^L, C_J^J)$ is stored in the list and $C_J^J$ as the answer is returned to $J$. The probability that $e(C_J^J, P)$ equals to $e(Q_J^1, K_{\mathrm{P}j})e(Q_J^2, K_{\mathrm{P}J}^{'})$ is at most $2^{-l}$ and negligible since $H(m^* \| \{K_{\mathrm{P}i}\}_{J\neq i} \| C_J^J)$ is uniformly distributed.

(5) ASign Queries: $J$ requests an ambiguous signature for any input in the form $(m^*, \{K_{\mathrm{P}i}\}_{i=1}^n, C_J^L)$ for published values $\{K_{\mathrm{P}i}\}_{i=1}^n$. $L$ checks the H-list for the existence of $m^*$. If $m^*$ does not exist, then $L$ calls the ASign algorithm to sign the message as usual. However, if $m^*$ exists, then $L$ simulates the signing oracle $\phi$ and selects $(v, a, r^*, a_2) \in Z_q^*$ at random. Further, the challenger $L$ sets $k_J^{'} = vP$ and $a_1 = r^* y - a_2\alpha$, and $L$ computes $Q_{J1}^* = a_1 P$ and $Q_{J2}^* = a_2 P$. Therefore, the challenger $L$ returns $(m^*, \{C_J^i\}_{i=1}^n, k_J^{'}, Q_{J1}^*, Q_{J2}^*)$ to the adversary.

(6) AVerify and Verify Queries: According to the verifying oracle $\phi$, $J$ is not allowed to query the verifying oracle $\phi$ for the challenge message, but $J$ can compute the forgery signature by himself.

(7) Output: The adversary produces a forgery signature $\sigma^* = (m^*, Q_{J1}^*, Q_{J2}^*, C_J^1, \cdots, C_J^n)$. According to the unforgeability, $J$ randomly selects an $r^* \in Z_q^*$, then the CDH problem can be solved successfully by computing $(r^*)^{-1} \bullet (Q_{J1}^* + \alpha Q_{J2}^*) = yP = \Phi$ with the probability:

$$\left( \frac{\mathrm{Adv}_{\mathrm{EF\text{-}CMA}}^J}{n-1} - \frac{q^{\Sigma} q^H + 1}{2^l} \right)^2 \leq \mathrm{Succ}_{\mathrm{CDH}}^J(l) \quad (2)$$

Therefore, if there is a non-negligible probability to generate the forgeable signature by the above game between the adversary and the challenger, then it will lead to a successful solution to the CDH problem.

*C. Ambiguity*

**Theorem 3** Our MPCS scheme is ambiguous and any outsiders couldn't determine the real signer until all individual keystones are released.

**Proof.** Due to the restriction of querying the verifying oracle $\phi$, the adversary $J$ can compute the signature by himself. If the adversary can query the verifying oracle $\phi$, $J$ can determine a bit $b \in \{0,1\}$ with the queries $(m^*, \sigma^*, K_{\mathrm{P}i_0})$ and $(m^*, \sigma^{'}, K_{\mathrm{P}i_1})$. $K_{\mathrm{P}i_0}$ and $K_{\mathrm{P}i_1}$ are two pairs of public keys from potential signers. $(\sigma^{'}, m^*, K_{\mathrm{P}i_b})$ will be accepted by the verifying oracle since $J$ can compute the new signature $\sigma^{'} = (Q_{J_1}^{'}, Q_{J_2}^{'}, C_J^1, \cdots, C_J^n)$ with $Q_{J_1}^{'} = Q_{J1}^* + K_{\mathrm{P}J}^{'}$ and $Q_{J_2}^{'} = Q_{J2}^* - K_{\mathrm{P}J}$. Besides, their signatures are ambiguous to the outsiders before users' own individual keystones are released.

As well known, the outsiders are not the designated receivers, thus if a participant $i \in \{1, \cdots, n\}$ is randomly selected from the participants and has generated a signature $\sigma_i$ of a message, the designated verifiers are convinced with the authenticity of the signature, while the outsiders aren't convinced that. Once the designated verifiers accept $\sigma_i$ from the signer $i$, they need to execute the second part of the Averify algorithm. If $\sigma_i$ passes the Averify algorithm successfully, then the designated verifiers can be convinced that $\sigma_i$ is indeed generated by the signer $i$. However, the outsiders maybe consider that the designated verifiers collude to generate the signature instead of the actual signer $i$. If the designated verifiers collude and collaboratively compute $Q_{i_1}^{'} = r^{'}P$, $Q_{i_2}^{'} = (k_{\mathrm{S}1} + \cdots + k_{\mathrm{S}i-1} + k_{\mathrm{S}i+1} + \cdots + k_{\mathrm{S}n})^{-1}(C_i^i - r^{'}K_{\mathrm{P}i})$, $r^{'} \in Z_q^*$, then the signature $\sigma_i = (Q_{i_1}^{'}, Q_{i_2}^{'}, C_i^1, \cdots, C_i^i, \cdots, C_i^n)$ will easily pass the Averify algorithm. That is $e(Q_{i_1}^{'}, K_{\mathrm{P}i})e(Q_{i_2}^{'}, \sum_{j\neq i} K_{\mathrm{P}j}) = e(C_i^i, P)$. Hence, the outsiders cannot distinguish the valid signature from any non-designated verifiers' viewpoint until the designated verifiers release their own individual keystones. Therefore, the signatures are ambiguous for the outsiders.

According to the above analysis, the ambiguity is defined by the simulation algorithms $D$ and $D^{'}$ between the adversary and the challenger.

(1) Assume $n$ pairs of keys $\{K_{\mathrm{P}j}, k_{\mathrm{S}j}\}_{j=1}^n$ are produced by the key generation algorithm. $J$ is fed with

$n-1$ public keys $\{K_{Pj}\}_{j=1}^{n-1}$ and $(K_{Pi_0}, k_{Si_0})$, $(K_{Pi_1}, k_{Si_1})$.

(2) The algorithm $D'$ flips a coin $b \in \{0,1\}$ and applies the ASign algorithm to generate the signature $\sigma^*$ of $m^*$, i.e., $\sigma^* = \mathrm{ASign}(m^*, k_{Si_b}, \{K_{Pj}\}_{j \neq i})$. Thus $D'$ sends $\sigma^*$ to $J$, and $J$ outputs a bit $b$.

(3) The simulation algorithm $D$ simulates the actions of the adversary $J$. Algorithm $D'$ simulates the actions of the challenger $L$. The public keys of two potential signers are $K_{Pi_0} = s_0 P$ and $K_{Pi_1} = s_1 P$ ($s_0, s_1 \in Z_q^*$). $D$ randomly chooses $a_j \in Z_q^*$ for $j \neq i$ and sets $K_{Pj} = a_j \gamma$ ($\gamma = xP$, $\psi = xyP$). Further, $D$ simulates the verifying oracle $\phi$. Once the signature $\sigma^*$ of $m^*$ is queried along with a bit $b$ to $\phi$, $D'$ browses the H-list to look for all forms $(m^*, \{K_{Pj}\}_{j=1}^n, k_i, C_i^j)$, sets $C_i^j = a_j \psi$ and randomly chooses $C_i^i \in G_1$ and $R \in G_1$. Then $D'$ tests whether $e(R, K_{Pj})$ is equal to $e(C_i^j, P)$ or not. If so, $D'$ picks an $r^* \in Z_q^*$ at random and computes $Q_{i_b}^{*2} = r^* P$ and $Q_{i_b}^{*1} = s_b^{-1}(C_i^i - r^* \sum_{j \neq i} K_{Pj})$. Thus $D$ outputs "accept" by the AVerify algorithm and picks a bit $b \in \{0,1\}$ at random, the adversary $J$ can randomly output $b$. The simulation is completely indistinguishable from the real game. Hence, the proposed MPCS scheme satisfies the property of ambiguity.

*D. Fairness*

**Theorem 4** Our MPCS scheme is fairness, since no matching signer left in a position where an individual keystone binds his signature to him while some of other signers' signatures are not bound to them after all participants released their own individual keystones.

**Proof.** If all signers are secure to implement the MPCS scheme, then one of the signers can get all their own individual keystones with the Extracting algorithm. If the MPCS scheme is unfair, by definition it must violate one of these two conditions with a non-negligible probability:

Case 1: One signer $j$ can obtain a valid signature such that $\sigma_j = (Q_j^1, Q_j^2, C_j^1, \cdots, C_j^n)$ is accepted by the Verify algorithm without getting a valid individual keystone $k_j$ from the other signers.

Then he/she can easily generate a forgery signature by the MPCS scheme. This implies an MPCS forgery, which contradicts the unforgeability of our MPCS scheme. Hence, the probability of the forgery signature is negligible due to the unforgeability.

Case 2: All signers can obtain their signatures $\sigma_i = (Q_i^1, Q_i^2, C_i^1, \cdots, C_i^n)$ for $i = 1, \cdots, n$ by the MPCS scheme. The signature $\sigma_j = (Q_j^1, Q_j^2, C_j^1, \cdots, C_j^n)$ from the signer $j$ is not accepted by the Verify algorithm while the other signatures are accepted by the Verify algorithm.

The signer $j$ is able to obtain a valid signature, since the signer can gain all individual keystones by himself/herself. Once one of the signers releases all their own individual keystones, their signatures can be bound simultaneously. If the other signers collude together, they can forge a valid signature instead of the actual signer $j$ and can easily compute $Q_{i_2}' = (k_{S1} + \cdots + k_{Si-1} + k_{Si+1} \cdots + k_{Sn})^{-1}(C_i^i - r' K_{Pi})$ and $Q_{i_1}' = r' P$. The signature can easily pass the AVerify algorithm. However, the actual signer $j$ has the right to check the validity of the individual keystone based on $e(K_{Pi}, k_j) = e(C_j^i, P)$, then the signer $j$ won't be left in an alone position. Therefore, our MPCS scheme satisfies the property of fairness.

In addition, the inherent unfairness of the previous concurrent signature schemes [12, 17, 18, 22] can be further avoided by adding a timestamp, if an initial signer doesn't release all individual keystones after $n$ signatures verified successfully for a long time. Once the time of releasing all keystones exceeds the limit, then the other signers have the same right to release the individual keystones instead of the initial signer, or to abandon the implementation of this MPCS scheme.

*E. Security*

**Theorem 5** Under the hardness of CDH problem, our MPCS scheme is secure in the random oracle model.

A secure MPCS scheme should satisfy the properties of correctness, unforgeability, ambiguity and fairness. Intuitively, our MPCS scheme satisfies the four properties, and the proof has been given from Theorem 1 to 4. Hence, our MPCS scheme is secure since CDH problem is hard to solve in the random oracle model.

*F. Performance comparison*

As for the computational overheads, we only consider bilinear mappings (denoted by $P$), multi-exponentiation on $G_2$ (denoted by $E$) and hash operation (denoted by $H$). The efficiencies of our MPCS scheme and Tonien's MPCS scheme are compiled in Table.1. From Table.1, our scheme uses less cost in the generation period of signature, and there aren't any bilinear and multi-exponentiation operations in the ASign algorithm. The complexity of our MPCS scheme is reduced apparently and more efficient than that of Tonien's MPCS scheme. Even if the length of our MPCS scheme is longer than Tonien's scheme, our scheme is better than Tonien's scheme due to the advantages in the ASIGN algorithm.

TABLE I

THE EFFICIENCIES OF TONIEN'S SCHEME AND OURS

| Algorithm | Tonien's MPCS scheme | Our MPCS scheme |
|---|---|---|
| Initial Sign | $2nH+(n-1)P+(n-1)E$ | $1H$ |
| Respond Sign | $2nH+(n-1)P+(n-1)E$ | $1H$ |
| Insider Verify | $2(n-1)H+(n+1)P+(n-1)E$ | $4(n-1)P+1H$ |
| Outsider Verify | $nH+2P$ | $nH+2P$ |

## V. CONCLUSIONS

A new MPCS scheme based on multi-designated verifiers is proposed. Unlike the previous schemes where the keystones are only released by the initial signer, all signers in our scheme have the same power to release all individual keystones once the time of releasing the individual keystones exceeds the limits. The efficiency and fairness of the MPCS scheme is improved further. The MPCS algorithm has a good scalability since it can be easily used not only between two signers but also among three or more signers. In other words, the scenarios of the concurrent signature can be easily applied from two-party to multi-party and vice versa, where the security assumption can be proved in the same way. The length of our signature scheme is linear with the increment of participants since the MPCS scheme is based on the concept of ring signature.

## ACKNOWLEDGMENT

This paper is supported by the National Natural Science Foundation of China (grant no. 61262084), the Natural Science Foundation of Jiangxi Province, China (grant nos. 20132BAB201019 and 20114BAB201018), the Research Foundation of the Education Department of Jiangxi Province (grant no. GJJ13057), and the Opening Project of Shanghai Key Laboratory of Integrate Administration Technologies for Information Security (grant no. AGK2012005).

## REFERENCES

[1] Z. W. Tan, "An off-line electronic cash scheme based on proxy blind signature," *Comput. J.* vol. 54, no. 4, pp. 505-512, 2011.

[2] O. Saebo, J. Rose, and J. Molka-Danielsen, "eParticipation: Designing and managing political discussion forums," *Soc. Sci. Comput. Rev.* vol. 28, no. 4, pp. 403-426, 2010.

[3] X. Y. Huang, Y. Mu, W. Susilo, W. Wu, J. Y. Zhou, and R. H. Deng, "Preserving transparency and accountability in optimistic fair exchange of digital signatures," *IEEE Trans. Inf. Foren. Sec.* vol. 6, no. 2, pp. 498-512, 2011.

[4] Q. Huang, D. C. S. Wong, and W. Susilo, "Group-oriented fair exchange of signatures," *Inform. Sciences*, vol. 181, no. 16, pp. 3267-3283, 2011.

[5] J. A. Garay, M. Jakobsson, and P. D. Mackenzie, "Abuse-free optimistic contract signing," In Proceedings of the 19th Annual International Cryptology Conference, CRYPTO '99. LNCS 1666, Springer, 1999, pp. 449-466.

[6] D. Bonech, and M. Naor, "Timed commitments (extended abstract). In: Proceedings of the 19th Annual International Cryptology Conference, CRYPTO 2000. LNCS 1880, Springer, 2000, pp. 236-254.

[7] S. D. Gordon, and J. Katz, "Partial fairness in secure two-party computation," *J. Cryptol.* Vol. 25, no.1, pp. 14-40, 2012.

[8] N. Asokan, V. Shoup, and M. Waidner, "Optimistic fair exchange of digital signatures," *IEEE J. Sel. Area. Comm.* Vol. 18, no. 4, pp. 593-610, 2000.

[9] X. X. Ye, Z. F. Cao, and X. H. Liang, "Fair document exchange protocol with confidentiality," *Comput. Eng.* Vol. 35, no. 4, pp. 149-151, 2009.

[10] Z. H. Shao, "Certificate-based fair exchange protocol of signature from pairings," *Comput. Netw.* Vol. 52, no. 16, pp. 3075-3084, 2008.

[11] Q. Shi, N. Zhang, and M. Merabti, "Fair exchange of valuable information: a generalised framework," *J. Comput. Syst. Sci.* vol. 77, no. 2, pp. 348-371, 2011.

[12] L. Q. Chen, C. Kudla, and K. G. Paterson, "Concurrent signatures," In Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, EUROCRYPT 2004. LNCS 3027, Springer, 2004, pp. 287-305.

[13] R. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," In Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security, ASIACRYPT 2001. LNCS 2248, Springer, 2001, pp. 552-565.

[14] J. Y. Hwang, "A note on an identity-based ring signature scheme with signer verifiability," *Theor. Comput. Sci.* vol. 412, no. 8-10, pp.796-804, 2011.

[15] W. Susilo, Y. Mu, and F. G. Zhang, "Perfect concurrent signature schemes," In Proceedings of the 6th International Conference on Information and Communications Security, ICICS 2004. LNCS 3269, Springer, 2004, pp. 14-26.

[16] G. L. Wang, F. Bao, and J. Y. Zhou, "The fairness of perfect concurrent signatures," In Proceedings of the 8th International Conference on Information and Communications Security, ICICS 2006. LNCS 4307, Springer, 2006, pp. 435-45.

[17] W. Susilo, and Y. Mu, "Tripartite concurrent signatures," In Proceedings of the 20th IFIP International Information Security Conference, Security and Privacy in the Age of Ubiquitous Computing. IFIP Advances in Information and Communication Technology 181, Springer, 2005, pp. 425-441.

[18] S. S. M. Chow, and W. Susilo, "Generic construction of perfect concurrent signatures," In Proceedings of the 7th International Conference on Information and Communications Security, ICICS 2005. LNCS 3783, , Springer, 2005, pp. 194-206.

[19] Z. J. Huang, K. F. Chen, and Y. M. Wang, "Analysis and improvements of two identity-based perfect concurrent signature schemes," *Informatica-Lithuan*, vol.18, no. 3, pp. 375-394, 2007.

[20] K. Nguyen, "Asymmetric concurrent signatures," In Proceedings of the 7th International Conference on Information and Communications Security, ICICS 2005. LNCS 3783, Springer, 2005, pp. 181-193.

[21] J. W. Liu, R. Sun, and K. Kwak, "Fair exchange signature schemes," *Sci China-Inform Sci.* vol. 53, no. 5, pp. 945-953, 2010.

[22] D. Tonien, W. Susilo, and R. Safavi-Naini, "Multi-party concurrent signatures," In Proceedings of the 9th

International Conference on Information Security, ISC 2006. LNCS 4176, Springer, 2006, pp. 131-145.

[23] C. Shieh, H. Lin, and S. Yen, "Fair multi-party concurrent signatures," In Proceedings of the 18th Cryptology and Information Security Conference, CISC 2008, pp. 108-118.

[24] L. L. Wang, "Multi-party concurrent signatures based on short ring signatures," In Proceedings of the IEEE International Conference on Wireless Communications, Networking and Information Security, WCNIS 2010, pp. 515-517.

[25] X. Tan, and Y. M. Zhao, "Unlinkable multi-party concurrent signatures," In Proceedings of the International Conference on Communications and Mobile Computing, CMC 2010, pp. 228-232.

[26] F. Laguilaumie, and D. Vergnaud, "Multi-designated verifiers signatures: anonymity without encryption," Inform. Process. Lett. Vol. 102, no. 2, pp. 127-132, 2007.

[27] B. Y. Kang, C. Boyd, and E. D. Dawson, "A novel identity-based strong designated verifier signature scheme," J. Syst. Software, vol. 82, no. 2, pp. 270-273, 2009.

[28] M. Jakobsson, K. Sako, and R. Impagliazzo, "Designated verifiers proofs and their applications," In Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, EUROCRYPT '96. LNCS 1070, Springer, 1996, pp. 142-154.

[29] W. Qin, and N. R. Zhou, "New concurrent digital signature scheme based on the Computational Diffie-Hellman problem," *J. China Univ. of Posts and Telecommun.*, vol. 17, no. 6, pp. 89-94, 2010.

[30] X. M. Hu, and S. T. Huang, "Secure identity-based blind signature scheme in the standard model," *J. Inf. Sci. Eng.* vol. 26, no. 1, pp. 215-230, 2010.

[31] D. Boneh, C. Gentry, and B. Lynn, "Aggregate and verifiably encrypted signatures from bilinear maps," In Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, EUROCRYPT 2003. LNCS 2656, Springer, 2003, pp. 416-432.