# Multi-Designated Verifiers Signatures Revisited

Sherman S. M. Chow
Department of Computer Science, Courant Institute of Mathematical Sciences
New York University, NY 10012, USA. (Email: schow@cs.nyu.edu)
*(Received Sept. 05, 2006; revised and accepted Oct. 06, 2006)*

## Abstract

Multi-Designated Verifier Signatures (MDVS) are privacy-oriented signatures that can only be verified by a set of users specified by the signer. We propose two new generic constructions of MDVS from variants of existing cryptographic schemes, which are ring signature from anonymous subset and multi-chameleon hash. We first devise a single add-on protocol which enables many existing identity-based (ID-based) ring signature schemes to support anonymous subset, which gives us three ID-based MDVS schemes. We then construct a multi-chameleon hash from an existing scheme with key exposure freeness. Interestingly, these two techniques can be seen as a multisignature version of Hess's ID-based signature and Schnorr signature respectively.

*Keywords: Bilinear pairings, chameleon hash, ID-based signature, key exposure, multi-chameleon hash, multi-designated verifiers signature, multisignature, privacy, ring signature, ring signature from anonymous subset*

## 1 Introduction

Designated verifier signature (DVS), introduced by Chaum [2] and Jakobsson *et al.* [14] independently, is a privacy-oriented signature scheme in which only a specific user can verify the signature produced. The "loss" of the non-repudiation property of traditional signature makes it useful in various commercial cryptographic applications, like call for tenders, electronic voting or electronic auction.

At CRYPTO 2003's rump session, Desmedt [10] asked for a Multi-Designated Verifiers Signature (MDVS) scheme, in which the number of designated verifiers is larger than one. Naturally, such feature helps the "multi-party version" of DVS's application, like distributed contract signing.

Subsequently, a generic MDVS scheme appeared in [16], which takes any discrete logarithm based ring signature scheme and any secure multi-party computation (SMC) protocol. The "designated" property of this generic MDVS comes from the fact that either the signer using his/her own private key, or the cooperation of all verifiers using SMC with their private keys as the input, can generate the signature. All verifiers get convinced about the real signer of the signature, but not any "outsiders". The use of SMC is crucial in their proposal, or the private key of each verifier is in risk if they really generate a "MDVS" cooperatively. In the shadow of this, one will generally believe that it is the signer but not the group of verifiers who generated the signature.

This situation is similar to the use of key-exposing chameleon hash function as pointed out by [1]. Chameleon hash function is a "public key hash function" which takes a message $m$ and a public key $pk$ to give a hashed value. Chameleon hash functions have the cryptographic properties of normal hash functions including pre-image resistance and collision-resistance (yet "limited"); but they are equipped with one additional property: it is easy to find collision and second-preimage with the knowledge of a secret key. If we use a chameleon hash function associated with the public key of the recipient to make the message digest, a single-designated verifier signature can be generated by using any digital signature scheme to sign on this message digest. The reason behind is that it is easy for the recipient to find collision with the help of his/her private key. Key-exposure problem here means making a collision exposes the associated private key, and hence every one is biased to believe that the single-designated verifier signature enabled by these classes of chameleon hash function usually comes from the signer instead of the verifier. Many chameleon hash functions (e.g. [15]) suffer from this problem.

Back to the MDVS case, if SMC is not used, the "designated" property of MDVS loses due to the key exposure property. SMC, while feasible, is costly (in terms of communication rounds, bandwidth, and computational complexity). Its complexity usually increases with the size of the group of the verifiers. Every one is biased to believe that the MDVS usually comes from the signer due to the uneven costs of MDVS generation by the signer and by group of verifiers, thus the real "designated" property of MDVS is in question.

We identify that if a ring signature scheme satisfies the "cooperative signing without key exposure" property, the requirement of SMC can be removed, which in turns make the idea of MDVS more realistic. To the best of author's knowledge, unfortunately, all existing discrete logarithm based ring signature schemes fail to satisfy this special

requirement by themselves. One of the results in this paper shows that: a ring signature from anonymous subset (i.e. 1-out-of-$n$-groups ring signature) can be derived if the ring signature scheme satisfies the "cooperative signing without key exposure" property.

Moreover, inspired by the generic construction of [16], we propose a new generic construction of multi-designated-verifiers signature scheme based on a modified notion of chameleon hash function: multi-chameleon hash. Informally, multi-chameleon hash is a chameleon hash function that forgery (i.e. finding collision) is only possible when all members of the group participate. Again, such class of schemes should be equipped with the "cooperative collision finding without key exposure" property.

### 1.1 Related Work

Ring signature gives 1-out-of-$n$ signer verifiability, allowing a user to sign anonymously on behalf of a group of $n$ spontaneously conscripted members. Rivest *et al.* first formalized the concept of ring signature in [20]. One of the most significant progresses in ring signature was given by [11], the anonymous identification scheme proposed enables a ring signature of constant-size (independently of the number of possible signers). With the help of an accumulator from bilinear pairings in [19], ID-based ring signatures of constant signature size are made possible [7, 19].

Ring signature scheme can be used to derive cryptographic schemes apart from multi-designated verifiers signature [16], like perfect concurrent signature like [6], and non-interactive deniable ring authentication [22].

The concept of designated verifier proof was introduced by Jakobsson *et al.* in 1996 [14], with the proposals of both interactive and non-interactive solution for a single verifier. After seven years, the non-interactive scheme was shown to be insecure by [23], where a dishonest signer can convince the designated recipient that a legitimate signature is issued while it is indeed not the case. Two countermeasures of the above attack were proposed in [23] as well.

Chameleon hash function was introduced in [15]. However, the key exposure problem was not addressed until 2004 in [3]. To the best of author's knowledge, [1] is the most recent work of chameleon hash function without key-exposure.

The first generic MDVS scheme was proposed in [16], which is based on ring signature scheme and implicitly relied on SMC as well. A concrete bi-designated verifiers signature scheme was proposed in [16] too.

### 1.2 Our Contributions

We give a bundle of results related to multi-designated verifiers signatures (MDVS). We identify that, without the help of secure multi-party computation (SMC), not all discrete logarithm based ring signature scheme can be used to construct the MDVS. We propose a technique that makes many ring signature schemes the suitable candidates for the generic construction of MDVS without the help of SMC. Our technique makes the cost of MDVS generation by the signer and by the group of verifiers more even. In other words, we increase the practicality of MDVS. Interestingly, the same technique can be applied to many ring signature schemes, which extended these ring signature schemes from anonymous single signer to anonymous set of signers. Consequently, three ID-based MDVS schemes are derived. As bonus results, our protocol extends existing ID-based ring signature schemes to support anonymous subset, and gives raise to a new ID-based multisignature, which are of independent interests.

We also propose a new notion of chameleon hash function called multi-chameleon hash, together with a concrete construction. From such a scheme, a new generic construction of multi-designated verifiers signature scheme is possible.

### 1.3 Organization

The rest of the paper is organized as follows. The next section contains the number-theoretic preliminaries of the cryptographic primitive and related complexity assumption used in the paper. The framework of MDVS and an existing generic construction, together with a discussion on the key exposure problem in cooperative signing using existing ID-based ring signature schemes, are given in Section 3. Section 4 discusses our proposed protocol for extending existing ID-based ring signature schemes to support anonymous subset, resulting three identity based multi-designated verifiers signature schemes. In Section 5, we propose our new generic construction of MDVS, together with our new building block, multi-chameleon hash function. An example showing how to modify an existing chameleon hash function to make it admissible for this new generic construction of MDVS is given in the same section too. Finally, Section 6 concludes the paper.

## 2 Cryptographic Primitive and Complexity Assumption

Bilinear pairing is an important primitive for many cryptographic schemes. In particular, many ring signatures schemes are pairing-based [5, 7, 8, 12, 13, 17, 19, 24]. Here, we describe some of its key properties.

Let $(\mathbb{G}_1, +)$ and $(\mathbb{G}_2, \cdot)$ be two cyclic groups of prime order $q$. The bilinear pairing is given as $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, which satisfies the following properties:

1) *Bilinearity*: For all $P, Q, R \in \mathbb{G}_1$, $\hat{e}(P + Q, R) = \hat{e}(P, R)\hat{e}(Q, R)$, and $\hat{e}(P, Q + R) = \hat{e}(P, Q)\hat{e}(P, R)$.

2) *Non-degeneracy*: There exists $P, Q \in \mathbb{G}_1$ such that $\hat{e}(P, Q) \neq 1$.

3) *Computability*: There exists an efficient algorithm to compute $\hat{e}(P,Q)$ $\forall P,Q \in \mathbb{G}_1$.

**Definition 1.** *Computational Diffie-Hellman (CDH) Problem: Let $\mathbb{G}_1$, and $P$ be as above. The CDH problem in $\mathbb{G}_1$ is as follows: Given $\langle P, aP, bP\rangle$ with $a,b,c \in \mathbb{Z}_q^*$, compute $abP \in \mathbb{G}_1$. An algorithm $\mathcal{A}$ has advantage $\epsilon$ in solving the CDH problem if $Pr[\mathcal{A}(\langle P, aP, bP\rangle) = abP] = \epsilon$. Here the probability is measured over random choices of $a, b$ in $\mathbb{Z}_q^*$ and the random bits of $\mathcal{A}$.*

# 3 Review of Multi-Designated Verifiers Signature

Multi-designated verifiers signature is defined as follows in [16].

**Definition 2.** *A Multi-Designated Verifiers Signature (MDVS) scheme consists of the following five polynomial time algorithms.*

- $\{params\} \leftarrow \mathsf{MDVS.Setup}(1^k)$*: A probabilistic algorithm that takes security parameter $k$ and outputs the public parameters $params$. In identity-based cryptosystem, it also produces a master secret key $msk$, kept in sercet by PKG (private key generator). For brevity, we omit the input of $params$ in the descriptions of other algorithms of MDVS.*
- $\{sk, pk\} \leftarrow \mathsf{MDVS.SKeyGen}(ID_S)$*: A probabilistic algorithm that takes an identity $ID_S$ and outputs a private key $sk_S$ together with the corresponding public key $pk_S$. For traditional public key infrastructure, the certificate authority (using the private key of the certificate authority) will issue a digital certificate binding the relationship of $ID$ and $pk$. For identity-based cryptosystem, no certificate is involved, and the PKG takes the additional input of master secret key $msk$ to generate the user's private key $sk$.*
- $\{sk, pk\} \leftarrow \mathsf{MDVS.VKeyGen}(ID_V)$*: A probabilistic algorithm that takes an identity $ID_V$ and outputs a private key $sk_V$ together with the corresponding public key $pk_V$. The differences between a scheme assuming traditional public key infrastructure or identity-based paradigm are similar as the* SKeyGen *algorithm above.*
- $\{\sigma\} \leftarrow \mathsf{MDVS.Sign}(m, sk, L)$*: A probabilistic/deterministic algorithm that takes a private key $sk$, a message $m$, a list $L$ that contains public keys of all designated verifiers and outputs a signature $\sigma$.*
- $\{\top \text{ or } \bot\} \leftarrow \mathsf{MDVS.Verify}(\sigma, m, L)$*: A deterministic algorithm that takes a message $m$, a signature $\sigma$, a public keys list $L$ (and possibly some subset of the private keys corresponding to $L$[1] ); which outputs either $\top$ or $\bot$ meaning accept or reject respectively, depending on whether $\sigma$ is a MDVS signature on the message $m$ with respect to the public key lists $L$.*

*These algorithms should satisfy the standard consistency and unforgeability requirement of a signature scheme. In addition, it should satisfy the* **source hiding** *property, i.e. given a message $m$ and a MDVS signature $\sigma$, it is (preferably unconditionally) infeasible to determine who, from the original signer or the whole group of the designated verifiers, performed the signing, even if the private keys of all parties are known.*

## 3.1 Review of Previous Generic Multi-Designated Verifiers Signature

We start by reviewing the definition of ring signature (e.g. see [20]).

**Definition 3.** *A ring signature scheme consists of the following polynomial time algorithms.*

- $\{params\} \leftarrow \mathsf{Ring.Setup}(1^k)$*: A probabilistic algorithm that takes security parameter $k$ and outputs the public parameters $params$ For brevity, we omit the input of $params$ in the descriptions of other algorithms below.*
- $\{sk, pk\} \leftarrow \mathsf{Ring.KeyGen}(ID)$*: A probabilistic algorithm that takes identity of the user $ID$ and outputs a private key $sk$ together with the corresponding public key $pk$.*
- $\{\sigma\} \leftarrow \mathsf{Ring.Sign}(sk, m, L)$*: A probabilistic algorithm that takes a private key $sk$, a message $m$ and a list $L$ that contains public keys including the one that corresponds to $sk$; outputs a ring signature $\sigma$.*
- $\{\top \text{ or } \bot\} \leftarrow \mathsf{Ring.Verify}(\sigma, m, L)$*: A deterministic algorithm that takes a message $m$ and a signature $\sigma$, and outputs either $\top$ or $\bot$ meaning accept or reject respectively, depending on whether $\sigma$ is the signature on message $m$ signed by private key $sk$ corresponding to one of the public key in the list $L$.*

*These algorithms should satisfy the standard consistency requirement, i.e. $\forall\{k, m, L\}$, if $\{sk, pk\} \leftarrow \mathsf{Ring.KeyGen}(1^k)$ and $\sigma \leftarrow \mathsf{Ring.Sign}(sk, m, L)$, then we must have the following equation satisfied: $\{\top\} \leftarrow \mathsf{Ring.Verify}(\sigma, m, L)$. Besides, it should satisfy the security requirements including existential unforgeability under adaptive chosen-message attack and signer ambiguity. Identity-based ring signature can be defined changing* Ring.Setup *and* Ring.KeyGen *accordingly as discussed in the review of MDVS. For full details, one may refer to [8].*

Here we review the generic construction of multi-designated verifiers signature from any discrete-logarithm based ring signature proposed in [16].

[1] Private key of the verifier is not necessary be the input if the scheme is essentially a proof showing either "the signer has signed on a message" or "I have the verifier's secret key" is true. The designated verifier, being confident that his/her private key is kept in secret, get convinced that the signer has signed on a message.

- MDVS.Setup = Ring.Setup.
- MDVS.SKeyGen = Ring.KeyGen, we denote $\{Q_S, S_S\}$ be the signer's public key and private key respectively.
- MDVS.VKeyGen = Ring.KeyGen, we denote $\{Q_{V_i}, S_{V_i}\}$ be the verifier $i$ 's public key and private key respectively.
- MDVS.Sign = Ring.Sign$(m, S_S, \{Q_S\} \cup \{\sum_{i=1}^{n} Q_{V_i}\})$.
- MDVS.Verify = Ring.Verify$(\sigma, m, \{Q_S\} \cup \{\sum_{i=1}^{n} Q_{V_i}\})$.

In short, it generates a 2-party ring signature. One party is the signer. Another is the "group of verifiers", with "group public key" $\sum_{i=1}^{n} Q_{V_i}$ and the "group private key" $\sum_{i=1}^{n} S_{V_i}$. The source hiding property comes from the following facts.

1) Either the private key of the signer or the "group private key" is used to generate the signature.
2) No one can distinguish which key (i.e. signer's private key or the "group private key") is actually used.
3) The group of verifiers can generate the "group private key" or complete the signing process without exposing their private keys (says by invoking a secure multi-party computation protocol).

The designated property comes from the following facts.

1) The scheme satisfies the source hiding property.
2) Any one in the group of verifiers, however, knows well that the signer generates the signature since it is necessary to use all the private keys of all verifiers to generate such a signature.

## 3.2 Applying the Generic Construction to Existing Schemes

Now we try applying the above generic construction to three existing identity-based ring signature schemes. The schemes below all use the same setup and private key generation algorithms.

Setup: The PKG randomly chooses $s \in_R \mathbb{Z}_q^*$, keeps it as the master secret key and computes the corresponding public key $P_{pub} = sP$. Let $H_1(\cdot)$ be a cryptographic hash function where $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1$. This hash function is for hashing any arbitrary identity into a value representing the user's public key. In addition, we require another cryptographic hash function $H_2(\cdot)$ for the signing of messages, where $H_2 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$. The system parameters are:

$$params = \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}(\cdot,\cdot), q, P, P_{pub}, H_1(\cdot)\, H_2(\cdot)\}.$$

KeyGen: The user with identity $ID \in \{0,1\}^*$ submits $ID$ to the PKG. The PKG sets the user's public key $Q_{ID}$ to be $H_1(ID) \in \mathbb{G}$, computes the user's private key $S_{ID}$ by $S_{ID} = sQ_{ID}$, where $s \in \mathbb{Z}_q^*$ is the master secret key of the PKG. Then PKG sends the private key to the user in a secure channel.

### 3.2.1 Notations

One can refer to the survey in [5] for a more detailed review of the schemes below. We will look into the the essential operation to generate a ring signature by the private key (the *close-the-ring* operation in the terminology of [5]) to see why they cannot satisfy the "cooperative signing without key exposure" property. To make the explanation easier, the case of only two designated verifiers ($ID_{V_1}$ and $ID_{V_2}$) is considered, and we name the purported originator of the message as $ID_S$. Let $L$ be the string $ID_{V_1}||ID_{V_2}||ID_S$. The terms in the form of $c_k$ are the commitments used in the signature. $A$ and $R_1$ are elements randomly chosen from $\mathbb{G}_1$.

### 3.2.2 Zhang and Kim 's Scheme [24]

Firstly, compute $c_1 = H_2(L||m||\hat{e}(A,P))$. Then, randomly choose elements $R_1$ from $\mathbb{G}_1$ and compute $c_2 = H_2(L||m||\hat{e}(R_1,P)\hat{e}(c_1 H_1(ID_S), P_{pub}))$. To give the signature, the verifiers need to compute $R_2 = A - c_2(S_{V_1} + S_{V_2})$. For verifier 1, it is easy to get the verifier 2's private key by computing ${c_2}^{-1}(A - R_2) - S_{V_1}$.

### 3.2.3 Lin and Wu 's Scheme [17]

We need to use an extra hash function $H_3 : \mathbb{G}_2 \rightarrow \mathbb{Z}_q$ in this scheme. Compute $c_1 = H_3(\hat{e}(A,P))$ and $c_2 = H_3(\hat{e}(R_1,P)\hat{e}(c_1 H_1(ID_S), P_{pub})^{H_2(m||L)})$. To give the signature, the verifiers need to compute $R_2 = A - H_2(m||L)c_2(S_{ID_{V_1}} + S_{ID_{V_2}})$. For verifier 1, it is easy to get the verifier 2's private key by computing $(c_2 H_2(m||L))^{-1}(A - R_2) - S_{ID_{V_1}}$.

### 3.2.4 Herranz and Sáez 's Scheme [12]

Compute $r_1 = \hat{e}(R_1, P)$ and $c_1 = H_2(L||m||r_1)$. To give the signature, the verifiers need to compute $U = c_1 H_1(ID_S)$, $r_2 = \hat{e}(A,P)\hat{e}(-P_{pub}, U)$, $c_2 = H_2(L||m||r_2)$ and $V = c_2(S_{ID_{V_1}} + S_{ID_{V_2}}) + A + R_1$. For verifier 1, it is easy to get the verifier 2's private key by computing ${c_2}^{-1}(V - A - R_1) - S_{ID_{V_1}}$.

# 4 Protocol for Supporting Anonymous Subset

Due to the key exposure problem in straightforward cooperative signing, the authors of [16] suggested the use of SMC. Now we propose the modification to remove such requirement by exploiting the structure of the ring signature scheme and the structure of the bilinear pairing.

### 4.1 Add-On Protocol

1) Each verifier $i$ randomly chooses an element $A_i$ from $\mathbb{G}_1$, computes $e_i = \hat{e}(A_i, P)$.

2) An arbitrary entity in the group of verifiers, after receiving these $e_i$s, computes $E = \prod_{i=1}^{n} e_i$.
($E$ is then used to replace the part $\hat{e}(A, P)$ in the above descriptions)

3) To close the ring, each verifier $i$ computes $\phi_i = A_i + \chi(S_{V_i})$.
($\chi$ is defined as $-c_2$, $-H_2(m||L)c_2$ and $c_2$ for [24], [17] and [12] respectively.)

4) An arbitrary entity in the group of verifiers, after receiving these $\phi_i$s, computes $\varphi = \sum_{i=1}^{n} \phi_i$ and proceeds the rest steps for giving a ring signature.
(Simply setting $R_2 = \varphi$ in [24] and [17] while computing $V = \varphi + R_1$ in [12]).

The above protocol exploits the additive homomorphic properties for the private keys and the corresponding public keys. Instead of only $n$ individual public keys, we can use $n$ "combined group public keys" in a ring signature. As a result, the underlying ID-based 1-out-of-$n$ ring signature schemes can be extended into ID-based 1-out-of-$n$-group ring signature schemes. The framework of ring signature from anonymous subset (i.e. 1-out-of-$n$-groups ring signature) is in the Appendix.

### 4.2 Security Analysis

For our protocol, we should consider the adversary to have the power to ask any group to compute the equation $\phi_i = A_i + \chi(S_{V_i})$ for any $\chi$ chosen by the adversary (which model the situation where the value of $\chi$ is modified by the adversary). Our protocol is insecure if it leaks any information about the private key $S_{V_i}$ that are useful in other parts of the protocol.

We analyze the security of our protocol by considering the below related multisignature scheme [18]. The adversary's power considered is chosen-message-and-group attacks, i.e. the adversary is given a signing oracle that can give signature on any message on behalf of any group. The adversary's goal is to give forgery that is not from these oracle queries, and there exists at least one user in the group that the adversary does not know, which means that information about the private key $S_{V_i}$ is leaked.

### 4.3 ID-based Multisignature Scheme

Multisignature schemes allow any subgroup of a group of users to sign a document jointly, such that a verifier is convinced that each member of the subgroup has participated in signing. Our add-on protocol can be used to implement an ID-based multisignature scheme as follows, which provdes a useful abstraction when we evaluate the security of our protocol.

Sign:

1) Each signer $ID_i$ randomly chooses an element $A_i$ from $\mathbb{G}_1$.

2) Each signer computes $r_i = \hat{e}(A_i, P)$, and broadcasts the value.

3) One of the signers combines these $r_i$s into $r = \prod_{i=1}^{n} \{r_i\}$, and this value is broadcasted.

4) Each signer $ID_i$ computes $c = H_2(m||r)$, where $m$ is the message to be signed.

5) Each signer $ID_i$ computes $R_i = cS_{ID_i} + A_i$.

6) One of the signers outputs the signature as $\sigma = (c, R = \sum_{i=1}^{n} \{R_i\})$.

Verify:
On receiving a message $m$ and a signature $\sigma = (c, R)$, the verifier accepts the signature if and only if $c = H_2(m||\hat{e}(P, R) \cdot \hat{e}(\sum_{i=1}^{n} \{H_1(ID_i)\}, -P_{pub})^c)$.

The scheme's correctness is easy to follow. For security proof, we relate the security of our protocol to the Computational Diffie-Hellman (CDH) problem. The challenger $\mathcal{C}$ will make use of the adversary $\mathcal{A}$ against the multisignature corresponding to our protocol to solve a random instance of CDH problem $(P, aP, bP)$.

1) *Simulation*: $\mathcal{C}$ firstly set $P_{pub} = bP$, and returns $r_i(aP)$ as the answer for the $i$-th identity-hashing query with probability $\zeta$ (which can be determined using the standard technique in [9]), and $r_iP$ with probability $1 - \zeta$. It is easy to see that $\mathcal{C}$ can answer any query relating to the second case, including private key generation queries. On the other hand, any private key generation queries for the first case will make $\mathcal{C}$ fails. Now we show how $\mathcal{C}$ can answer the signing queries for $ID_i$.

   $\mathcal{C}$ randomly chooses $R_i \in \mathbb{G}_1$ and $h \in \mathbb{Z}_q^*$, computes $r_i = \hat{e}(P_{pub}, Q_{ID_{V_i}})^{-h}\hat{e}(P, R_i)$. Notice that if the combined commitment $c = H_2(m||r) = h$, $(c, R_i)$ is a partial signature since $c = H_2(m||\hat{e}(P, R_i) \cdot \hat{e}(-P_{pub}, Q_{ID_{V_i}})^c)$. For $c = H_2(m||r) = h$ to hold, the simulation uses the rewinding technique in the proof of the unforgeability of the multisignature scheme in [18].

2) *Solving CDH Problem*: A forgery is successful if it is a multisignature that involves the signature on the message $m_i$ signed by $ID_i$, where the adversary has not asked for $ID_i$'s private key and made no signing query to get this signature on $m_i$. If the adversary has chosen to forge the signature of $ID_i$ that is defined as $r_i(aP)$ by $\mathcal{C}$, then we can solve CDH problem with the help of $\mathcal{A}$.

   Similar to what is done in the forking lemma, algorithm $\mathcal{C}$ solves the CDH problem by replaying algorithm $\mathcal{A}$ with the same random tape but different choices of $H_2$ to obtain two valid signatures $(c, R)$

and $(c', R')$, and the solution of CDH problem can be obtained by $abP = (r_i(c - c'))^{-1}(R - R')$.

### 4.4 Robustness and Efficiency

If this protocol can be deviated easily (e.g. any single entity in the group can unnoticeably deviate from the protocol to make the final output is invalid), the designated property of MDVS is also questionable. In our protocol, the robustness can be ensured easily by checking whether the equality $\hat{e}(\phi_i, P) = e_i\hat{e}(Q_{V_i}, P_{pub})^{\chi}$ holds.

Each party only needs to perform one pairing operation, one point exponentiation and one point multiplication, while these steps are essential for the case of a single signer to generate the MDVS as well. The arbitrary entity in Step 2 and Step 4 of the protocols only needs to do $n-1$ additions in $\mathbb{G}_1$ and $n-1$ multiplications in $\mathbb{G}_2$, where $n$ is the number of designated verifiers. Notice that addition in $\mathbb{G}_1$ and multiplication in $\mathbb{G}_2$ are much more efficient than the dominating operation: bilinear pairing. Besides, our protocol only takes two rounds and no secure channel is necessary.

## 5 A New Generic Multi-Designated Verifiers Signature

Before proposing our new generic construction of multi-designated verifiers signature, we first review the definition of chameleon hash function, which is the basic form of our proposed building block: multi-chameleon hash.

**Definition 4.** *A chameleon hash function consists of the following four polynomial time algorithms.*

- $\{params\} \leftarrow \mathsf{Chameleon.Setup}(1^k)$*: A probabilistic algorithm that takes security parameter k and outputs the public parameters params. For brevity, we omit the input of params in the descriptions of other algorithms below.*
- $\{sk, pk\} \leftarrow \mathsf{Chameleon.KeyGen}$*: A probabilistic algorithm that outputs a private key sk together with the corresponding public key pk, for each user.*
- $\{h\} \leftarrow \mathsf{Chameleon.Hash}(pk, m, \tau)$*: A probabilistic algorithm that takes a public key pk, a message m and a random factor $\tau$, outputs a hash value h.*
- $\{\tau'\} \leftarrow \mathsf{Chameleon.Forge}(sk, m, m', \tau)$*: An efficient probabilistic algorithm that takes a message m, a random factor $\tau$ and another message $m'$, outputs $\tau'$ such that the equality* $\mathsf{Chameleon.Hash}(pk, m, \tau) = \mathsf{Chameleon.Hash}(pk, m', \tau')$ *holds.*

*These algorithms should satisfy the following security requirements.*

1) *Collision resistance: Given the input of pk, m, $\tau$ and $m'$, it is computationally infeasible to find $\tau'$ such that* $\mathsf{Chameleon.Hash}(pk, m, \tau) = \mathsf{Chameleon.Hash}(pk, m', \tau')$ *without the secret key sk.*
2) *Semantic security: Let $H[X]$ be the entropy of a random variable $X$ and $H[X|Y]$ be the entropy of the variable $X$ given the value of a random function $Y$ of $X$, the conditional entropy $H[m|C]$ of the message $m$ given its chameleon hash value $C$ equals the total entropy $H[m]$ of the message space.*
3) *Key exposure freeness: Given a polynomially many queries on oracle access to* Chameleon.Forge*, there is no efficient algorithm that can find a collision that has not been queried.*

### 5.1 Framework of Multi-Chameleon Hash Function

A multi-chameleon hash function consists of the following six polynomial time algorithms: $\mathsf{Setup}$, $\mathsf{KeyGen}$, $\mathsf{Single-Hash}$, $\mathsf{Single-Hash}$, $\mathsf{Multi-Hash}$, and $\mathsf{Multi-Forge}$.

- $\{params\} \leftarrow \mathsf{Setup}(1^k)$: On a unary string input $1^k$ where $k$ is a security parameter, it produces the common public parameters $params$.
- $\{sk, pk\} \leftarrow \mathsf{KeyGen}$: A probabilistic algorithm that outputs a private key $sk$ together with the corresponding public key $pk$ for each user.
- $\{h\} \leftarrow \mathsf{Single-Hash}(pk, m, \tau)$: A probabilistic algorithm that takes a public key $pk$, a message $m$ and a random factor $\tau$, outputs a hash value $h$.
- $\{\tau'\} \leftarrow \mathsf{Single-Forge}(sk, m, m', \tau)$: An efficient probabilistic algorithm that takes a message $m$, a random factor $\tau$ and another message $m'$, outputs $\tau'$ such that the equality $\mathsf{Single-Hash}(pk, m, \tau) = \mathsf{Single-Hash}(pk, m', \tau')$ holds.
- $\{h\} \leftarrow \mathsf{Multi-Hash}(\cup_{i=1}^{n}\{pk_i\}, m, \tau)$: On input of $n$ public keys, a message $m$ and a random factor $\tau$, it outputs a hash value $h$ corresponding to the input parameters.
- $\{\tau'\} \leftarrow \mathsf{Multi-Forge}(\cup_{i=1}^{n}\{sk_i\}, m, m', \tau)$: On input of $n$ private keys, a message $m$, a message $m'$ different from $m$, the hash value $h$, and a random factor $\tau$, it outputs another random factor $\tau'$ such that $\mathsf{Multi-Hash}(\cup_{i=1}^{n}\{pk_i\}, m, \tau) = \mathsf{Multi-Hash}(\cup_{i=1}^{n}\{pk_i\}, m', \tau')$, where the $n$ private keys used are the corresponding private keys of $\cup_{i=1}^{n}\{pk_i\}$.

The security requirements of $\mathsf{Single-Hash}$ and $\mathsf{Single-Forge}$ are the same as those in a normal chameleon hash. Semantic security is about the hash value so the same applies on both $\mathsf{Single-Hash}$ and $\mathsf{Multi-Hash}$. $\mathsf{Multi-Hash}$ and $\mathsf{Multi-Forge}$ should satisfy the following security requirements.

1) *Collision resistance:* Given the input of $\cup_{i=1}^{n}\{pk_i\}$, $m$, $\tau$ and $m'$, it is computationally infeasible to find $\tau'$ such that $\mathsf{Hash}(\cup_{i=1}^{n}\{pk_i\}, m, \tau) = \mathsf{Hash}(\cup_{i=1}^{n}\{pk_i\}, m', \tau')$ without all of the secret keys corresponding to $\cup_{i=1}^{n}\{pk_i\}$.

2) *Key exposure freeness:* Given a polynomially many queries on oracle access to $\mathsf{Multi}-\mathsf{Forge}$, there is no efficient algorithm that can find a collision that has not been queried.

## 5.2 Proposed Construction

We firstly review the scheme proposed in [1], then we show our modification to extend it into a multi-chameleon hash function, with security analysis.

- $\{params\} \leftarrow \mathsf{Chameleon.Setup}(1^k)$: We need a normal cryptographic hash function $H_4$ to map arbitrary length inputs to bit strings of fixed length.
- $\{sk, pk\} \leftarrow \mathsf{Chameleon.KeyGen}$: Let $p$ be a $k$-bit safe-prime such that $p = 2q + 1$ where $q$ is also prime. Suppose $g$ is a generator of the subgroup of quadratic residues $\mathbb{Q}_p$ of $\mathbb{Z}_p^*$ of order $q$. Private key $sk$ is randomly chosen from $\{2, \cdots, q-1\}$ and the corresponding public key $pk$ is $y = g^{sk}$.
- $\{h\} \leftarrow \mathsf{Chameleon.Hash}(pk, m, \{r, s\})$: Suppose $\{r, s\} \in \mathbb{Z}_q^2$, compute $e = H_4(m, r)$ and $h = r - (y^e g^s \mod p) \mod q$.
- $\{r', s'\} \leftarrow \mathsf{Chameleon.Forge}(sk, m, m', \{r, s\})$: Let $h = \mathsf{Chameleon.Hash}(m, \{r, s\})$ where $e = H_4(m, r)$. Randomly chooses $k'$ from $\{1, 2, \cdots, q-1\}$, computes $r' = h + (g^{k'} \mod p) \mod q$, $e' = H_4(m', r')$, and $s' = k' - e'(sk) \mod q$.

To extend the above scheme to multi-chameleon hash, we make the following changes.

1) Each verifier $i$ randomly chooses an element $k_i'$ from $\{1, 2, \cdots, q-1\}$, computes $g_i = g^{k_i'}$.
2) An arbitrary entity in the group of verifiers, after receiving these $g_i$s, computes $r' = h + \prod_{i=1}^{n} g_i \mod p$ and $e' = H_4(m', r')$.
3) Each verifier $i$ computes $\phi_i = k_i' - e' sk_i \mod q$.
4) An arbitrary entity in the group of verifiers, after receiving these $\phi_i$s, computes $s' = \sum_{i=1}^{n} \phi_i \mod q$.

## 5.3 Generic Construction of MDVS

Given a discrete logarithm based signature scheme which is existentially unforgeable against adaptive chosen message attack ($\{\mathsf{DigSig.Setup}, \mathsf{DigSig.Sign}, \mathsf{DigSig.Verify}\}$) and a discrete logarithm based multi-chameleon hash function with "cooperative collision finding without key exposure" property, we can construct a multi-designated verifiers signature scheme as follows.

- $\mathsf{MDVS.Setup} = \mathsf{Chameleon.Setup}, \mathsf{DigSig.Setup}$.
- $\mathsf{MDVS.SKeyGen} = \mathsf{DigSig.KeyGen}$, suppose $\{Q_S, S_S\}$ is the signer's public key and private key respectively.
- $\mathsf{MDVS.VKeyGen} = \mathsf{Chameleon.KeyGen}$, suppose $\{Q_{V_i}, S_{V_i}\}$ is the verifier $i$ 's public key and private key respectively.
- $\mathsf{MDVS.Sign} = (\mathsf{DigSig.Sign}$ $(S_S, \mathsf{Chameleon.Multi}-\mathsf{Hash}(\sum_{i=1}^{n} Q_{V_i}, m, \{r, s\}))$, $m, \{r, s\})$, i.e. signing the chameleon hash, and including the message with the random factor such that the chameleon hash value can be reconstructed.
- $\mathsf{MDVS.Verify} = \mathsf{DigSig.Verify}$ $(\sigma, \mathsf{Chameleon.Multi}-\mathsf{Hash}(\sum_{i=1}^{n} Q_{V_i}, m, \{r, s\}), Q_S)$ where $\sigma$ is the signature generated by $\mathsf{DigSig.Sign}$.

In short, it generates a chameleon signature [1] where the recipient is a "group of verifiers", with "group public key" $\sum_{i=1}^{n} Q_{V_i}$ and the "group private key" $\sum_{i=1}^{n} S_{V_i}$. The source hiding property comes from the following facts.

1) Either the private key of the signer or the "group private key" is used to generate the signature.
2) No one can distinguish which key (i.e. signer's private key or the "group private key") is actually used since both parties can generate this signature.
3) The group of verifiers can forge a collision with the input of the "group private key" or by using $\mathsf{Multi}-\mathsf{Forge}$ without exposing their private keys.

The designated property comes from the following facts.

1) The scheme satisfies the source hiding property.
2) Any one in the group of verifiers, however, knows well that the signer generates the signature since it is necessary to use all the private keys of all verifiers to generate such a signature.

## 5.4 Discussion

Note that the proposed scheme inherits the "weakness" of chameleon signatures, in which the group of verifiers can only forge a signature on a message on their wish only if they have obtained a signature on some other message from the signer already. Nevertheless, we think that it is not a great problem even they cannot generate such a signature spontaneously. Once a signature is presented by the signer, the group of verifiers can forge another message, so any third party can only know the signer has signed on some messages, but do not know exactly which one.

Similar to our add-on protocol in previous section, our extension to chameleon-hash function can be considered as the multisignature extension [18] of Schnorr's signature [21]. As a result, our extension's security can be proved in a similar way as the proof in [18].

## 6 Conclusion

Existing construction of multi-designated verifier signature (MDVS) scheme relies on the fact that either the actual signer or the group of verifiers can create the same signature. However, for the group of verifiers to create the signature, they may leak their private key due to the failure of existing ring signature schemes in supporting the "cooperative signing without key exposure" requirement. Secure multi-party computation (SMC) protocol is one of the solutions to remedy but incurs an uneven cost of MDVS generation between the signer and the group of verifiers. As a result, one is easily biased to believe that the MDVS usually comes from the signer and hence the real "designated" property of MDVS loses.

By exploiting the structure of the ring signature schemes, we find that a generic SMC can be replaced by a simple protocol that is yet robust and efficient. Our proposed protocol can be applied to many identity-based ring signature schemes and converted them into ring signature scheme supporting anonymous subset. As a result, three practical identity-based multi-designated verifiers signature schemes are devised. We also proposed a new generic construction of multi-designated verifiers signature from a new notion called multi-chameleon hash functions: chameleon hash functions that satisfy the "cooperative collision finding without key exposure" property.

We leave it as an open problem to devise other (generic) constructions of multi-designated verifiers signature and other multi-chameleon hash function. Another interesting direction is to study the generic construction of multi-designated verifiers signature with stronger privacy guarantee, like the use of encryption and ring signature in the construction of [4].

## References


[1] G. Ateniese, "On the key exposure problem in chameleon hashes," *Security in Communication Networks, 4th International Conference, SCN 2004*, LNCS 3352, pp. 165–179, C. Blundo and S. Cimato, Editors, Springer-Verlag, Amalfi, Italy, Sep. 8-10, 2004, Revised Selected Papers, Amalfi, Italy, 2005.

[2] D. Chaum, "Private signature and proof systems," *United States Patents*, no. 5, pp. 493-614, 1996.

[3] X. Chen, F. Zhang, and K. Kim, "Chameleon hashing without key exposure," *Proceedings of Information Security, 7th International Conference, ISC 2004*, LNCS 3225, pp. 87-98, K. Zhang and Y. Zheng, Editors, Springer-Verlag, Palo Alto, CA, USA, Sep. 27-29, 2004.

[4] S. S. M. Chow, "Identity-based strong multi-designated verifiers signatures," *Proceedings of the Public Key Infrastructure, Third European PKI Workshop: Theory and Practice, EuroPKI 2006*, LNCS 4043, pp. 257-259, A. S. Atzeni and A. Lioy, Editors, Springer-Verlag, Turin, Italy, June 19-20, 2006.

[5] S. S. M. Chow, R. W. C. Lui, L. C. K. Hui, and S. M. Yiu, "Identity based ring signature: Why, how and what next," *Public Key Infrastructure, Second European PKI Workshop: Research and Applications, EuroPKI 2005*, LNCS 3545, pp. 144-161, D. W. Chadwick and G. Zhao, Editors, Springer-Verlag, Canterbury, UK, Jun. 30- Jul. 1, 2005, Revised Selected Papers, 2005.

[6] S. S. M. Chow and W. Susilo, "Generic construction of (identity-based) perfect concurrent signatures," *Proceedings of the Information and Communications Security, 7th International Conference, ICICS 2005*, LNCS 3783, pp. 194-206, S. Qing, W. Mao, J. Lopez, and G. Wang, Editors, Springer-Verlag, Beijing, China, Dec. 10-13, 2005.

[7] S. S. M. Chow, W. Susilo, and T. H. Yuen, "Escrowed linkability of ring signatures and its applications," *Proceedings of the Vietcrypt 2006, First International Conference on Cryptology*, LNCS 4341, pp. 175–192, P. D. Dieu and P. Q. Nguyen, Editors, Springer-Verlag, Vietnam, Hanoi, Vietnam, Sep. 25-28, 2006.

[8] S. S. M. Chow, S. M. Yiu, and L. C. K. Hui, "Efficient identity based ring signature," *Applied Cryptography and Network Security, Third International Conference, ACNS 2005*, LNCS 3531, pp. 499-512, J. Ioannidis, A. D. Keromytis, and M. Yung, Editors, Springer-Verlag, New York, USA, Jun 7-10, 2005.

[9] J. S. Coron, "On the exact security of full domain hash," *Proceedings of Cryptology - Crpto '00, 20th Annual International Cryptology Conference*, LNCS 1880, pp. 229-235, M. Bellare, Editor, Springer-Verlag, Santa Barbara, California, USA, Aug. 20-24, 2000.

[10] Y. Desmedt, "Verifier-designated signatures," 2006. (http://web.archive.org/web/20060904033040/www.cs.fsu.edu/∼desmedt/lectures/verifier-designated-signatures.pdf)

[11] Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup, "Anonymous identification in ad hoc groups," *Advances in Cryptology - Eurocrypt '04, International Conference on the Theory and Applications of Cryptographic Techniques*, LNCS 3027, pp. 609-626, C. Cachin and J. Camenisch, Editors, Springer-Verlag, Interlaken, Switzerland, May 2-6, 2004.

[12] J. Herranz and G. Sáez, "New identity-based ring signature schemes," *Proceedings of the Information and Communications Security, 6th International Conference, ICICS 2004*, LNCS 3269, pp. 27-39, J. Lopez, S. Qing, and E. Okamoto, Editors, Springer-Verlag, Malaga, Spain, Oct. 27-29, 2004.

[13] F. Hess, "Efficient Identity Based Signature Schemes based on Pairings," *Selected Areas in Cryptography, 9th Annual International Workshop, SAC 2002*, LNCS 2595, pp. 310-324, K. Nyberg and H. M. Heys, Editors, Springer-Verlag, St. John's, Newfoundland, Canada, Aug. 15-16, 2002.

[14] M. Jakobsson, K. Sako, and R. Impagliazzo, "Designated verifier proofs and their applications," *Advances in Cryptology - Eurocrypt '96, International Conference on the Theory and Application of Cryptographic Techniques*, LNCS 1070, pp. 143-154, U. M. Maurer, Editor, Springer-Verlag, Saragossa, Spain, May 12-16, 1996.

[15] H. Krawczyk and T. Rabin, "Chameleon hashes," *Network and Distributed System Security Symposium (NDSS) 2000*, pp. 143-154, 2000.

[16] F. Laguillaumie and D. Vergnaud, "Multi-designated verifiers signatures," *Proceedings of the Information and Communications Security, 6th International Conference, ICICS 2004*, LNCS 3269, pp. 495-507, J. Lopez, S. Qing, and E. Okamoto edit, Springer-Verlag, Malaga, Spain, Oct. 27-29, 2004.

[17] Chih-Yin Lin and Tzong-Chen Wu, "An identity-based ring signature scheme from bilinear pairings," *Cryptology ePrint Archive*, Report 2003/117, 2003. Available at http://eprint.iacr.org

[18] S. Micali, K. Ohta, and L. Reyzin, "Accountable-subgroup multisignatures: Extended abstract," *Proceedings of the CCS '01: 8th ACM conference on Computer and Communications Security*, pp. 245-254, New York, NY, USA, ACM Press, 2001.

[19] L. Nguyen, "Accumulators from bilinear pairings and applications," *Topics in Cryptology - CT-RSA 2005, The Cryptographers' Track at the RSA Conference 2005*, LNCS 3376, pp. 275-292, A. J. Menezes Editor, Springer-Verlag, San Francisco, CA, USA, Feb. 14-18, 2005.

[20] R. L. Rivest, A. Shamir, and Y. Tauman, "How to Leak a Secret," advances in *Cryptology - Asiacrypt '01, 7th International Conference on the Theory and Application of Cryptology and Information Security*, LNCS 2248, pp. 552-565, C. Boyd, Editor, Springer-Verlag, Gold Coast, Australia, Dec. 9-13, 2001.

[21] C. P. Schnorr, "Efficient signature generation by smart cards," *Journal of Cryptology: The Journal of the International Association for Cryptologic Research*, vol. 4, no. 3, pp. 161-174, 1991.

[22] W. Susilo and Y. Mu, "Non-interactive deniable ring authentication," *Information Security and Cryptology - ICISC 2003, 6th International Conference*, LNCS 2971, pp. 386-401, J. I. Lim and D. H. Lee, Editors, Springer-Verlag, Seoul, Korea, Nov. 27-28, 2003, Revised Papers, Seoul, Korea, 2004.

[23] G. Wang, "An attack on not-interactive designated verifier proofs for undeniable signatures," *Cryptology ePrint Archive*, Report 2003/243, 2003. Available at http://eprint.iacr.org.

[24] F. Zhang and K. Kim, "ID-based blind signature and ring signature from pairings," *Advances in Cryptology - Asiacrytp '02, 8th International Conference on the Theory and Application of Cryptology and Information Security*, LNCS 2501, pp. 533-547, Y. Zheng, Editor, Springer-Verlag, Queenstown, New Zealand, Dec. 1-5, 2002.

# Appendix

## ID-Based Ring Signature from Anonymous Subset

An ID-based ring signature from anonymous subset consists of four algorithms: Setup, KeyGen, Sign, and Verify.

- Setup: On a unary string input $1^k$ where $k$ is a security parameter, it produces the master secret key $s$ and the common public parameters $params$, which include a description of a finite signature space and a description of a finite message space.
- KeyGen: On an input of signer's identity $ID \in \{0,1\}^*$ and the master secret key $s$, it outputs the signer's secret signing key $S_{ID}$. (The corresponding public verification key $Q_{ID}$ can be computed easily by everyone.)
- Sign: On input of a message $m$, $n$ group of users' identities $\{\mathcal{U}_i\}$, where $1 \leq i \leq n$ and $\mathcal{U}_i = \{ID_{i_j}\}$, and the secret keys $\{S_{ID_{s_j}}\}$ of all members of one of the group $\mathcal{U}_s$, where $1 \leq s \leq n$; it outputs an ID-based ring signature from anonymous subset $\sigma$ on the message $m$.
- Verify: On input of a ring signature $\sigma$, a message $m$ and $n$ group of users' identities $\{\mathcal{U}_i\}$, where $1 \leq i \leq n$ and $\mathcal{U}_i = \{ID_{i_j}\}$, it outputs $\top$ for "true" or $\bot$ for "false", depending on whether $\sigma$ is a valid signature signed by all members of a certain group in the $\{\mathcal{U}_i\}$ on a message $m$.

These algorithms must satisfy the standard consistency constraint of ID-based ring signature from anonymous subset, i.e. if $\sigma = \mathsf{Sign}(m, \{\mathcal{U}_i\}, \{S_{ID_{s_j}}\})$, we must get "true" from the verification algorithm taking the signature, the message and the groups of identities as the input, i.e. $\mathsf{Verify}(\sigma, \{\mathcal{U}_i\}, m) = \top$.

For a secure ID-based ring signature from anonymous subset, we need unforgeability and signer ambiguity.

The following EUF-IDRSAS-CMIA2 game played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ formally defines the *existential unforgeability of ID-based ring signature under adaptive chosen-message-and-identity attack*.

*EUF-IDRSAS-CMIA2 Game*:

*Setup*: The challenger $\mathcal{C}$ takes a security parameter $k$ and runs the Setup to generate common public parameters $params$ and the master secret key $s$. $\mathcal{C}$ sends $params$ to $\mathcal{A}$.

*Attack*: The adversary $\mathcal{A}$ can perform a polynomially bounded number of queries in an adaptive manner (that is, each query may depend on the responses to the previous queries). The types of queries allowed are described below.

- Hash functions queries: $\mathcal{A}$ can ask for the values of the hash functions (e.g. $H_1(\cdot)$ and $H_2(\cdot)$ in our proposed scheme) for any input.

- KeyGen: $\mathcal{A}$ chooses an identity $ID$. $\mathcal{C}$ computes $\mathsf{KeyGen}(ID) = S_{ID}$ and sends the result to $\mathcal{A}$.
- Sign: $\mathcal{A}$ chooses $n$ group of users' identities $\{\mathcal{U}_i\}$, where $1 \leq i \leq n$ and $\mathcal{U}_i = \{ID_{i_j}\}$, and any message $m$. $\mathcal{C}$ outputs an ID-based ring signature from anonymous subset $\sigma$.

*Forgery*: The adversary $\mathcal{A}$ outputs an ID-based ring signature $\sigma$ and $n$ group of users' identities $\{\mathcal{U}_i\}$, where $1 \leq i \leq n$ and $\mathcal{U}_i = \{ID_{i_j}\}$. The only restriction is that $(m, \{\mathcal{U}_i\})$ does not appear in the set of previous Sign queries and for each group of identities $\{\mathcal{U}_i\}$, at least one secret key in $\{S_{ID_{i_j}}\}$ is never returned by any KeyGen query. It wins the game if $\mathsf{Verify}(\sigma, \{\mathcal{U}_i\})$ is equal to $\top$. The advantage of $\mathcal{A}$ is defined as the probability that it wins.

**Definition 5.** *An ID-based ring signature scheme from anonymous subset has the existential unforgeability against adaptive chosen-message-and-identity attacks property (EUF-IDRSAS-CMIA2 secure) if no adversary has a non-negligible advantage in the EUF-IDRSAS-CMIA2 game.*

**Definition 6.** *An ID-based ring signature scheme from anonymous subset has the unconditional group of signers ambiguity if for any n group of users' identities* $\{\mathcal{U}_i\}$, *where* $1 \leq i \leq n$ *and* $\mathcal{U}_i = \{ID_{i_j}\}$, *any message m and any signature* $\sigma$, *where* $\sigma = \mathsf{Sign}(m, \{\mathcal{U}_i\})$; *any verifier* $\mathcal{A}$ *not from the actual signer group, even with unbounded computing resources, cannot identify the actual group of signers with probability better than a random guess. That is,* $\mathcal{A}$ *can only output the actual signers group indexed by s with probability no better than* $\frac{1}{n}$.

**Sherman S. M. Chow** is currently a PhD candidate in the Courant Institute of Mathematical Sciences at New York University. He has been a research intern of Fuji Xerox Palo Alto Laboratory and a visiting scholar of the Information Security Institute at Queensland University of Technology. In summer 2008, he is a research intern of Crypto and Anti-Piracy Group, Microsoft Research.

He has published over 30 papers in the area of identity-based cryptography, certificateless cryptography, two-factor encryption, key agreement, group-oriented signature, and distributed system security (e.g. e-voting, e-cash, P2P, privacy-preserving queries). He has also served on the program committee of ProvSec '07, ACIS '06 (as a program co-chairman), and as reviewers for many conferences and journals including TCC '08, ISC '08, Crypto '07, ACM E-Commerce '07, Eurocrypt '06, Asiacrypt '05, DKE and JUCS.