

University of Wollongong  
**Research Online**

---

Faculty of Engineering and Information  
Sciences - Papers: Part A

Faculty of Engineering and Information  
Sciences

---

1-1-2012


## Multi-level controlled signature

Pairat Thorncharoensri  
pt78@uow.edu.au

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

Yi Mu  
*University of Wollongong*, ymu@uow.edu.au

Follow this and additional works at: <https://ro.uow.edu.au/eispapers>

 Part of the [Engineering Commons](#), and the [Science and Technology Studies Commons](#)

---

### Recommended Citation

Thorncharoensri, Pairat; Susilo, Willy; and Mu, Yi, "Multi-level controlled signature" (2012). *Faculty of Engineering and Information Sciences - Papers: Part A*. 153.  
<https://ro.uow.edu.au/eispapers/153>

Research Online is the open access institutional repository for the University of Wollongong. For further information contact the UOW Library: [research-pubs@uow.edu.au](mailto:research-pubs@uow.edu.au)

---

## Multi-level controlled signature

### Abstract

In this work, we present and establish a new primitive called Multi-level Controlled Signature. This primitive allows a signer to specify a security level to limit the verifiability of the signature. This primitive works as follows. Without losing generality, we assume the security levels of a group of users are defined in ascending order, where "A" represents the lowest security level and "Z" represents the highest security level, respectively. When a signer signs a message by specifying a security level "C", all users who have a security level greater than "C" will be able to verify while other users whose security levels are "A", "B" or "C" cannot verify the authenticity of this message. This primitive resembles some similarities with other existing primitives, such as Hierarchical Identitybased Encryption/Signatures, policy-based cryptography, but we stress that this primitive is unique in the sense that other primitives cannot satisfy all requirements as stated above efficiently. In this paper, we develop a security model for such a primitive. We present two concrete constructions that are proven secure in our model. The first scheme has a constant signature size, while the second scheme is more efficient in terms of verifier's private information. We provide a comparison between our schemes and illustrate where each scheme is applicable in the real world scenario.

### Keywords

controlled, signature, multi, level

### Disciplines

Engineering | Science and Technology Studies

### Publication Details

Thorncharoensri, P., Susilo, W. & Mu, Y. (2012). Multi-level controlled signature. *Lecture Notes in Computer Science*, 7690 (2012), 96-110.

# Multi-level Controlled Signature

Pairat Thorncharoensri<sup>1,2</sup>, Willy Susilo<sup>1\*</sup> and Yi Mu<sup>1</sup>

<sup>1</sup> School of Computer Science & Software Engineering, University of Wollongong, Australia  
pt78@uowmail.edu.au, {wsusilo, ymu}@uow.edu.au

<sup>2</sup> Department of Information Systems, Faculty of Business, City University of Hong Kong, Hong Kong  
pthornch@cityu.edu.hk

**Abstract.** In this work, we present and establish a new primitive called *Multi-level Controlled Signature*. This primitive allows a signer to specify a security level to limit the verifiability of the signature. This primitive works as follows. Without losing generality, we assume the security levels of a group of users are defined in ascending order, where “A” represents the lowest security level and “Z” represents the highest security level, respectively. When a signer signs a message by specifying a security level “C”, all users who have a security level greater than “C” will be able to verify while other users whose security levels are “A”, “B” or “C” cannot verify the authenticity of this message. This primitive resembles some similarities with other existing primitives, such as Hierarchical Identity-based Encryption/Signatures, policy-based cryptography, but we stress that this primitive is unique in the sense that other primitives cannot satisfy all requirements as stated above efficiently. In this paper, we develop a security model for such a primitive. We present two concrete constructions that are proven secure in our model. The first scheme has a constant signature size, while the second scheme is more efficient in terms of verifier’s private information. We provide a comparison between our schemes and illustrate where each scheme is applicable in the real world scenario.

## 1 Introduction

Multi-level marketing (MLM) is a marketing strategy that is aimed at compensating promoters of selling companies not only for product sales they personally generate, but also for the sales of others they introduced to the company. Consider a scenario where Alice, who is a representative at the MLM company, would like to inform the amount of sale to her “upline” distributors (meaning that all distributors who are higher than her level). In this situation, Alice would like to convince only people in the “upline” connection, rather than everyone who has a level below her. In this situation, Alice’s signature on the message must be “designated” to everyone whose level is higher than her.

Note that at the first glance, the notion of designated verifier signature, as introduced in [10], might be used to solve this situation by allowing Alice to sign her message multiple times designated to *all* members who have higher level than her. Nevertheless, in this situation, Alice may not know who are the other members whose level are higher than her (other than the one who is directed above her level, i.e. the person who introduced her to the company). Hence, the notion of designated verifier signature cannot really solve this problem.

According to the best of our knowledge, there exists no cryptographic primitives that can be used to solve the above problem efficiently. Therefore, in this work, we introduce the notion of *multi-level controlled signatures* to solve the aforementioned problem. We present the security model to capture this notion, together with two concrete schemes. Our first scheme benefits from its constant signature size, and this scheme is suitable for an organization where there are many security levels involved. The second scheme benefits from the short credential involved and therefore, it is suitable for an organization where the security levels involved are not widely spread.

---

\* This work is supported by the ARC Future Fellowship (FT0991397).

## 1.1 Related Work

In this section, we will review some work in the literature that are closely related to our proposed notion.

The notion of policy-based cryptography was put forth by Bagga and Molva in [1]. This notion includes policy-based encryption schemes and policy-based signature schemes. In the policy-based signature schemes, a signer can only sign a message if he/she satisfies the required policy. Policy-based signatures ensure the security of the scheme from the signer’s point of view, namely message integrity, authenticity and non-repudiability. In policy-based signatures, any verifier can verify the authenticity of the message, but nobody can forge the authenticity of the signatures that have been signed by a signer that satisfies a stated policy. This is in contrast to multi-level controlled signatures. In multi-level controlled signatures, the policy will guard the verifiers, and hence, only verifiers who satisfy some policies will be able to verify the authenticity of the message.

The other related notion is the notion of Hierarchical Identity-based Encryption/Signature. The Hierarchical Identity-based Encryption (HIBE) system [8, 9, 3, 5] is a unified concept between a hierarchy system and Identity-based Encryption (IBE) system [11, 4] where an identity at level  $k$  of the hierarchical system can issue a private key for its descendant identity, but it cannot decrypt a message on behalf of other identity except its descendants. At the first sight, it seems quite straightforward to construct multi-level controlled signatures from HIBE and a standard signature scheme. This is done as follows. First, a public key generator (PKG) in HIBE assumes the role of a trusted authority (TA) in the scheme. PKG generates HIBE private keys as credentials for verifiers. Second, a signer  $S$  generates the signature by signing on a concatenation of the actual message  $M$  and some uniformly random dummy message  $M_R$  in the plaintext space of the HIBE scheme. Next, by using  $ID$  as the security level,  $S$  uses HIBE to encrypt  $M_R$  with  $ID$  as his/her public key. Finally, the multi-level controlled signature consists of the signature on  $M||M_R$  and the HIBE ciphertext of  $M_R$ . Without having the credentials allowing to decrypt the HIBE ciphertext and to obtain  $M_R$ , the signature cannot be verified. The above generic construction seems to be correct, but unfortunately there is a big drawback to the scheme. If a verifier who holds the valid credentials (and hence, allowing him/her to decrypt the HIBE ciphertext) exposes  $M_R$  with the above signature, then everyone can verify this signature without the need of having the credentials to decrypt the HIBE ciphertext. This contradicts with the security model of the multi-level controlled signature, which will be presented in the Section 3. Essentially, this is the basic security requirement of the multi-level controlled signature to avoid someone from exposing some information that will be affecting the security of the scheme. Hence, it is clear that the above generic construction scheme is indeed insecure in the required security model.

The Hierarchical Identity-based Signature (HIBS) scheme [8, 6, 9] is a natural conversion from the HIBE scheme. Similar to the HIBE, the ancestor identity of the hierarchical system can issue a private key for its descendant identity. However, it cannot sign on a message on behalf of other identities except its descendants. The purpose of HIBE systems is to reduce the bottleneck in a large network, where the PKG of the IBE system is applied, and to limit the scope of key escrow. For the HIBS, it is, however, similar to policy-based signature schemes where it only provides for the signer in the message integrity, and the authenticity and non-repudiation but not the authorization for the verifier.

The other work related to this notion is the designated verifier signature proofs, as mentioned earlier. It was introduced by Jakobsson, Sako and Impagliazzo in [10]. A proof does not provide only authentication of a message but it also provides the deniability property that allows the signer to deny the signature (since the verifier can also generate such a proof). Hence, only the designated verifier can verify the proof on the message.

## 1.2 Our Contributions

In this paper, we introduce the notion of multi-level controlled signature (MLCS) schemes and present two concrete constructions of MLCS schemes. The notion of MLCS scheme allows only receivers, who hold a credential for a certain security level specified by the sender (or signer), to

verify the authenticity of the signed message. Furthermore, we define and formalize the notion of MLCS scheme and its security model. Our concrete constructions are proven secure in our model.

### Organization of The Paper

The paper is organized as follows. In the next section, we review some preliminaries that will be used throughout this paper. In the Section 3, we introduce the definition of MLCS and its security notions. Next, the first concrete scheme together with its security proof will be provided in Section 4. Then, in the Section 5, we will provide the second concrete scheme and its security proof. Finally, the comparison of two concrete schemes and conclusion of the paper will be presented in the last section.

## 2 Preliminaries

### 2.1 Notation

Throughout this paper, the following notations will be used. When we say that a function  $f : \mathbb{N} \rightarrow \mathbb{R}$  is *negligible*, it means that, for all constant  $c > 0$  and for all sufficiently large  $n$ ,  $f(n) < \frac{1}{n^c}$ . Let  $poly(\cdot)$  be a deterministic polynomial function. For all polynomials  $poly(k)$  and for all sufficiently large  $k$ , we say that  $q$  is polynomial-time in  $k$  if  $q \leq poly(1^k)$ . Let  $l \stackrel{\$}{\leftarrow} L$  denote the operation of picking  $l$  at random from a (finite) set  $L$ .

### 2.2 Bilinear Pairing

We denote by  $\mathbb{G}_1$  and  $\mathbb{G}_2$  cyclic multiplicative groups. Their generators are  $g_1$  and  $g_2$ , respectively. Let  $p$  be a prime and the order of both generators. Let  $\mathbb{G}_T$  be another cyclic multiplicative group with the same order  $p$ . We denote by  $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$  a bilinear mapping with the following properties:

1. *Bilinearity*:  $\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$  for all  $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2, a, b \in \mathbb{Z}_p$ .
2. *Non-degeneracy*: There exists  $g_1 \in \mathbb{G}_1$  and  $g_2 \in \mathbb{G}_2$  such that  $\hat{e}(g_1, g_2) \neq 1$ .
3. *Computability*: There exists an efficient algorithm to compute  $\hat{e}(g_1, g_2)$  for all  $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2$ .

Note that there exists  $\varphi(\cdot)$  function which maps  $\mathbb{G}_1$  to  $\mathbb{G}_2$  or vice versa in one time unit.

### 2.3 Complexity Assumptions

**Definition 1 (Computation Diffie-Hellman (CDH) Problem).** *Given a 3-tuple  $(g, g^x, g^y \in \mathbb{G}_1)$  as input, output  $g^{x \cdot y}$ . An algorithm  $\mathcal{A}$  has advantage  $\epsilon'$  in solving the CDH problem if*

$$\Pr[\mathcal{A}(g, g^x, g^y) = g^{x \cdot y}] \geq \epsilon'$$

where the probability is over the random choice of  $x, y \in \mathbb{Z}_q^*$  and the random bits consumed by  $\mathcal{A}$ .

**Assumption 1. Computation Diffie-Hellman Assumption [7, 2]** We say that the  $(t, \epsilon')$ -CDH assumption holds if no PPT algorithm with time complexity  $t(\cdot)$  has advantage at least  $\epsilon'$  in solving the CDH problem.

**Definition 2 (Decision Bilinear Diffie-Hellman (DBDH) Problem).** *Given a random 4-tuple  $(g, g^x, g^y, g^z) \in \mathbb{G}_1$  and a random integer  $Z \in \mathbb{G}_T$  as input, decide whether or not  $Z = \hat{e}(g, g)^{xyz}$ . An algorithm  $\mathcal{A}$  is said to  $(\mathfrak{t}, \epsilon')$  solves the DBDH problem in  $\mathbb{G}_1, \mathbb{G}_T$ , if  $\mathcal{A}$  runs in time  $\mathfrak{t}$ , and*

$$|\Pr[\mathcal{A}(g, g^x, g^y, g^z, Z = \hat{e}(g, g)^{xyz}) = 1] - \Pr[\mathcal{A}(g, g^x, g^y, g^z, Z = \hat{e}(g, g)^d) = 1]| \geq \epsilon',$$

where the probability is taken over the random choices of  $x, y, z, d \in \mathbb{Z}_p, g \in \mathbb{G}_1$ , and the random bits consumed by  $\mathcal{A}$ .

**Assumption 2. Decision Bilinear Diffie-Hellman Assumption** We say that the  $(t, \epsilon')$ -DBDH assumption in  $\mathbb{G}_1, \mathbb{G}_T$  holds if there is no PPT algorithm with time complexity  $t(\cdot)$  has advantage at least  $\epsilon'$  in solving the DBDH problem.

### 3 Multi-level Controlled Signature Schemes (MLCS)

#### Model

Let  $TA$  be a trusted authority who issues credentials associated with a security level in the multi-level security system. In multi-level controlled signature schemes, there are three main players called a signer, a verifier and a trusted authority  $TA$ . A signer  $S$  generates a signature, which can be verified *only* by a verifier  $V$  who holds a credential satisfying the multi-level security policy.  $TA$  is responsible to issue a credential for  $V$ . Let  $A_V$  denote a security level in the multi-level security policy. We define  $MP$  to be a multi-level security policy which contains a policy that indicates a level of security clearance of the verifier. Without losing generality, we assume that the order of the security levels is increasing, for example, the higher number is the higher security level<sup>3</sup>. Let  $MP = "A_V > n"$  where  $n$  is the number indicated the security level. Generally, we can use other type of index or symbol to indicate the security level. In the following, we provide a definition of multi-level controlled signature scheme as follows.

**Definition 3.** A multi-level controlled signature scheme  $\Sigma$  is an 6-tuple ( $Setup, TKeyGen, SKeyGen, CreGen, Sign, Verify$ ) such that

#### Signature Scheme Setup:

- *System Parameters Generation (Setup):*  
On input a security parameter  $K$ , a PPT algorithm named  $Setup$  outputs the system parameters  $param$ . That is,  $param \leftarrow Setup(1^K)$ .
- *TA Key Generator (TKeyGen) :*  
On input the system parameters  $param$ , a PPT algorithm named  $TKeyGen$  outputs strings  $(sk_{TA}, pk_{TA})$  where they denote a secret key and a public key of trusted authority, respectively. That is,  
 $\{pk_{TA}, sk_{TA}\} \leftarrow TKeyGen(param)$ .
- *Signer Key Generator (SKeyGen) :*  
On input the system parameters  $param$ , a PPT algorithm named  $SKeyGen$  outputs strings  $(sk_S, pk_S)$  where they denote a secret key and a public key of a signer, respectively. That is,  $\{pk_S, sk_S\} \leftarrow SKeyGen(param)$ .
- *Verifier Credential Generator (CreGen) :*  
On input the system parameters  $param$ , the  $TA$ 's public key, and an assertion  $A_V$  indicated a security level of verifier, a PPT algorithm named  $CreGen$  outputs a credential for verifier  $VCR$  That is,  $VCR \leftarrow CreGen(param, pk_{TA}, sk_{TA}, A_V)$ .

#### Multi-level Controlled Signature Signing ( $Sign$ ):

$Sign$  is a PPT algorithm that, on input the system parameters  $param$ , the trusted authority's public key  $pk_{TA}$ , the signer's secret key  $sk_S$ , the signer's public key  $pk_S$ , a message  $M$  and the multi-level security policy  $MP$ , it outputs signer's signature  $\sigma$ . That is,  $\sigma \leftarrow Sign(param, M, sk_S, pk_S, pk_{TA}, MP)$ .

#### Multi-level Controlled Signature Verification ( $Verify$ ):

$Verify$  is an algorithm that, on input the system parameters  $param$ , the trusted authority's public key  $pk_{TA}$ , the signer's public key  $pk_S$ , the multi-level security policy  $MP$ , a credential  $VCR$ , a message  $M$  and a signature  $\sigma$ , it outputs a verification decision  $d \in \{Accept, Reject\}$ . That is,  $d \leftarrow Verify(param, M, \sigma, pk_{TA}, pk_S, MP, VCR)$ .

#### 3.1 Security Model

Before modelling the ability of the adversaries in breaking the security of MLCS schemes, we first describe the oracles used in the security model as follows:

**SSO oracle :** An adversary  $\mathcal{A}$  can make at most  $q_S$  query to  $SSO$  for signatures  $\sigma$  on its choice of a message  $M$ .  $SSO$  outputs a response by running the  $Sign$  algorithm to generate a signature  $\sigma$  on a message  $M$  corresponding with  $pk_{TA}$ ,  $pk_S$  and  $MP$ . Then,  $SSO$  returns  $\sigma$  to  $\mathcal{A}$ .

<sup>3</sup> We note that for a decreasing order security levels, our scheme can be modified trivially.

**$\mathcal{VCO}$  oracle :**  $\mathcal{A}$  can make at most  $q_C$  query to  $\mathcal{SSO}$  for credentials  $VCR$  corresponding to a security level  $A_V$ .  $\mathcal{VCO}$  responds  $\mathcal{A}$  with a corresponding credential  $VCR$ .

**$\mathcal{VSO}$  oracle :** By giving  $\sigma, M$  to  $\mathcal{VSO}$  as inputs,  $\mathcal{A}$  can make at most  $q_V$  query for the verification of a signature  $\sigma$ .  $\mathcal{VSO}$  returns with **Accept** or **Reject** depending on the validation of signature  $\sigma$  and the message  $M$ .

**Unforgeability** The unforgeability property of MLCS schemes is to prevent an attacker, who has an access to the credential oracle, to generate a new multi-level controlled signature  $\sigma_*$  on a message  $M^*$ . Formally, the unforgeability in this model provides an assurance that one, with an access to  $\mathcal{SSO}$  oracle,  $\mathcal{VCO}$  oracle, the signer's public parameters  $pk_S$  and the trusted authority's public key, should be unable to produce a new multi-level controlled signature on a message  $M^*$  even with arbitrarily chosen multi-level security policy  $MP$ , a message  $M$  and the entire credentials as inputs.

Let  $CM-A$  denote the adaptive chosen message and credential exposure attack and let  $EUF-MLCS$  denote the existential unforgeability of MLCS scheme. The following experiment between the adversary  $\mathcal{A}_{EUF-MLCS}^{CM-A}$  and a simulator  $\mathcal{F}$  models a security against existential unforgeability under the adaptive chosen message and credential exposure attack.

With an adaptive strategy,  $\mathcal{A}$  arbitrarily makes queries to  $\mathcal{SSO}$  and  $\mathcal{VCO}$  oracle on its choice of a message  $M$ . Let  $\overline{VCR}$  be the credentials for the entire security level. After the queries process, assume that  $\mathcal{A}$  outputs a forged signature  $\sigma_*$  on a message  $M^*$  with respect to the public key  $pk_S$  and multi-level security policy  $MP^*$ .  $\mathcal{A}$  wins the experiment if:

1.  $Accept \leftarrow Verify(M^*, \sigma_*, pk_S, MP^*, \overline{VCR})$ .
2.  $\sigma_*$  is not the output of that  $\mathcal{A}$  previously made a request to  $\mathcal{SSO}$  oracle.

Let  $Succ_{EUF-MLCS}^{CM-A}(\cdot)$  be the success probability function of that  $\mathcal{A}_{EUF-MLCS}^{CM-A}$  wins the above experiment.

**Definition 4.** A MLCS scheme is  $(\mathfrak{t}, q_H, q_S, q_C, \epsilon)$ -secure existentially unforgeable under a chosen message and credential exposure attack if there are no PPT adversary  $\mathcal{A}_{EUF-MLCS}^{CM-A}$  such that the success probability  $Succ_{EUF-MLCS}^{CM-A}(k) = \epsilon$  is negligible in  $k$ , where  $\mathcal{A}_{EUF-MLCS}^{CM-A}$  runs in time at most  $\mathfrak{t}$ , makes at most  $q_H$  hash queries,  $q_S$  signing queries, and  $q_C$  verification queries.

**Coalition-resistance** In this section, we will describe the coalition-resistant property of MLCS schemes. This property is to prevent an attacker, as a group of corrupted credential holders (verifiers), to verify a multi-level controlled signature  $\sigma_*$  on a message  $M^*$  with a multi-level security policy  $MP$ , in which the attacker does not have a credential satisfied the security level indicated in  $MP$ .

Let  $CR-MLCS$  denote the existential coalition-resistance of MLCS scheme. Let  $\mathcal{A}_{CRI-PCS}^{CMP-A}$  be the adaptively chosen message and chosen multi-level security policy distinguisher and let  $\mathcal{F}$  be a simulator. The following game between  $\mathcal{F}$  and  $\mathcal{A}$  describes the existential coalition-resistance of MLCS scheme under a chosen message and chosen multi-level security policy attack. We divide the game into two phases and run them as follows:

1. **Phase 1 :** With any adaptive strategies,  $\mathcal{A}$  arbitrarily issues a request of queries to  $\mathcal{SSO}$  and  $\mathcal{VCO}$  oracles. The oracles response as per their design.
2. **Challenge :** After the first phase,  $\mathcal{A}$  outputs  $M^*$  and  $MP^* = "A_V \geq l"$  such that:
  - a. On input  $MP^*$  and  $M^*$ ,  $\mathcal{A}$  never issued a request for a multi-level controlled signature to  $\mathcal{SSO}$  queries.
  - b. With  $MP^* = "A_V \geq l"$ ,  $\mathcal{A}$  can issue a request of credential to  $\mathcal{VCO}$  queries for a security level  $A_V < l$ .

If the above condition is satisfied,  $\mathcal{F}$  chooses a random bit  $b \xleftarrow{\$} \{0, 1\}$ . If  $b = 1$  then, on input a multi-level security policy  $MP^*$  and a message  $M^*$ ,  $\mathcal{F}$  issues a request for a multi-level controlled signature to  $\mathcal{SSO}$  queries. Then  $\mathcal{F}$  responds  $\mathcal{A}$  with  $\sigma^*$  as an output from  $\mathcal{SSO}$

queries. Otherwise, on input a multi-level security policy  $MP^*$ , a message  $M^*$ , a valid policy-controlled signature  $\sigma$  on message  $M^*$  with policy  $MP^*$  and a credentials  $VCR$ ,  $\mathcal{F}$  computes a (simulated) invalid multi-level controlled signature  $\sigma^*$ . Then  $\mathcal{F}$  responds  $\mathcal{A}$  with  $\sigma^*$ .

3. **Phase 2** : In this phase,  $\mathcal{A}$  can arbitrarily return to *Phase 1* or *Challenge*. With one condition, at least one set of challenges  $M^*, MP^*, \sigma^*$  must be valid and satisfy the condition in the challenge phase.
4. **Guessing** :  $\mathcal{A}$  finally outputs a guess  $b'$  based on a challenge  $M^*, MP^*, \sigma^*$ . The distinguisher wins the game if  $b = b'$ .

We denote by  $Succ_{CR-MLCS}^{CMP-A}(\cdot)$  the success probability function of that  $\mathcal{A}_{CR-MLCS}^{CMP-A}$  wins the above experiment.

**Definition 5.** A MLCS scheme is  $(\mathfrak{t}, q_H, q_S, q_C, \epsilon)$ -secure existentially coalition-resistance under a chosen message and chosen multi-level security policy attack if there are no PPT distinguisher  $\mathcal{A}_{CR-MLCS}^{CMP-A}$  such that the success probability  $Succ_{CR-MLCS}^{CMP-A}(k) = |\Pr[b = b'] - \Pr[b \neq b']| = \epsilon$  is negligible in  $k$ , where  $\mathcal{A}_{CR-MLCS}^{CMP-A}$  runs in time at most  $\mathfrak{t}$ , make at most  $q_H$  hash queries,  $q_S$  signing queries, and  $q_C$  verification queries.

## 4 The First MLCS Scheme

In this section, we present our first concrete construction of MLCS schemes. Let  $H : \{0, 1\}^* \rightarrow \mathbb{G}_1$  be a collision-resistant hash function. Let  $h : \{0, 1\}^* \rightarrow \mathbb{Z}_p^*$  be a collision-resistant hash function. Let  $\mathbb{G}_1$  and  $\mathbb{G}_T$  denote two groups of prime order  $p$ . Let  $\hat{e}$  be the bilinear mapping function, which maps  $\mathbb{G}_1$  to  $\mathbb{G}_T$ . The above mapping function is defined as  $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ . The scheme is described as follows.

*Setup* : On input a security parameter  $\mathcal{K}$ , a trusted third party randomly chooses a prime  $p \approx \text{poly}(1^\mathcal{K})$ . Choose a random generator  $g \in \mathbb{G}_1$  and a bilinear mapping function  $\hat{e}$ . Select two hash functions  $H(\cdot)$  and  $h(\cdot)$ . Let us denote by  $\text{param} = (p, \hat{e}, g, H, h)$  the system parameters. Then, *Setup* returns  $\text{param}$ .

*TKeyGen* : Let  $n$  be a number of security levels. On input a system parameters  $\text{param}$ , a trusted authority  $TA$  randomly generates a private key  $sk_{TA}$  and a public key  $pk_{TA}$  for each security level as follows: select random integers  $\mu_1, \dots, \mu_n, \gamma_1, \dots, \gamma_n, a, b, c_1, \dots, c_n \in \mathbb{Z}_p$ . Let  $pk_{TA} = (U_1 = g^{\mu_1}, \dots, U_n = g^{\mu_n}, W_1 = g^{\gamma_1}, \dots, W_n = g^{\gamma_n}, \mathbb{A} = g^a, \mathbb{B} = g^b)$  denote a public key. Then, *TKeyGen* returns  $sk_{TA} = (\mu_1, \dots, \mu_n, \gamma_1, \dots, \gamma_n, a, b, c_1, \dots, c_n)$  as a private key of the trusted authority and  $pk_{TA} = (U_1, \dots, U_n, W_1, \dots, W_n, \mathbb{A}, \mathbb{B})$  as a public key of the trusted authority.

*SKeyGen* : On input a system parameters  $\text{param}$ , a signer  $S$  randomly generates a private key  $sk_S$  and a public key  $pk_S$  as follows. First, choose a random integer  $x \in \mathbb{Z}_p$ . Let  $\mathbb{X} = g^x$ ;  $\mathbb{W} = \mathbb{A}^x$ ;  $\mathbb{U} = \mathbb{B}^x$  denote a public key. Then, *SKeyGen* set  $sk_S = x$  as a private key of the signer and  $pk_S = (\mathbb{X}, \mathbb{W}, \mathbb{U})$  as a public key of the signer. Finally, *SKeyGen* returns  $sk_S, pk_S$ .

*CreGen* : Let  $A_V$  indicate a security level of verifier, for example,  $A_V = \text{"D"}$ . On input a system parameters  $\text{param}$ , the trusted authority's public key  $pk_{TA}$ , the trusted authority's private key  $sk_{TA}$  and a security level of verifier  $A_V = l$  that verifier is satisfied to obtain, a trusted authority  $TA$  randomly generates a set of credential strings  $VCR = (CV_1, \dots, CV_l, CR_1, \dots, CR_l)$ , where  $i$  is an index of security level, as follows:  $TA$  randomly selects  $\nu_1, \dots, \nu_l \in \mathbb{Z}_p^*$  and computes each credential  $CV_i = g^{c_i \cdot \nu_i}$ ;  $CR_i = g^{(\mu_i \cdot \gamma_i - \mu_{i-1} \cdot \gamma_{i-1} - a \cdot c_i \cdot \nu_i) / (b)}$  and then returns  $VCR = (CV_1, \dots, CV_l, CR_1, \dots, CR_l)$  to the verifier as a credential for a security level assertion  $A_V = l$ . The verifier checks the validity of each  $CV_i$  and  $CR_i$  as follows:

$$\hat{e}(U_i, W_i) \stackrel{?}{=} \hat{e}(\mathbb{A}, CV_i) \hat{e}(\mathbb{B}, CR_i) \hat{e}(U_{i-1}, W_{i-1}).$$

*Sign* : Given  $\text{param}, pk_{TA}, sk_S, pk_S, MP = \text{"}A_V \geq l\text{"}$  and a message  $M$ ,  $S$  computes a multi-level controlled signature  $\sigma$  on a message  $M$  as follows:

$$\begin{aligned} r, k &\stackrel{\$}{\leftarrow} \mathbb{Z}_p, \sigma_1 = g^r, \sigma_2 = \mathbb{X}^r, \sigma_3 = \mathbb{W}^r, \sigma_4 = \mathbb{U}^r, \\ \Gamma &= \sigma_1 \parallel \sigma_2 \parallel \sigma_3 \parallel \sigma_4 \parallel pk_S \parallel pk_{TA} \parallel MP, \sigma_5 = g^k, \sigma_6 = H(\Gamma)^x, \\ \sigma_7 &= h(\hat{e}(U_l, W_l)^{x \cdot r}) + h(M \parallel \Gamma \parallel \sigma_5), \sigma_8 = k + \sigma_7 \cdot x. \end{aligned}$$



The multi-level controlled signature on a message  $M$  is  $\sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6, \sigma_7, \sigma_8)$ .  $S$  publishes  $M, \sigma, MP$ .

*Verify* : Let  $VCR = CV_1, \dots, CV_l, CR_1, \dots, CR_l$ , be a set of credentials that verifier possessed. Parse  $\Gamma = \sigma_1 || \sigma_2 || \sigma_3 || \sigma_4 || pk_S || pk_{TA} || MP$ . Given  $pk_S, pk_{TA}, pk_V, VCR, MP = "A_V \geq l"$ ,  $\sigma$  and a message  $M$ , a verifier  $V$  checks whether

$$\begin{aligned} \hat{e}(\sigma_1, \mathbb{X}) &\stackrel{?}{=} \hat{e}(\sigma_2, g), \quad \hat{e}(\sigma_3, g) \stackrel{?}{=} \hat{e}(\sigma_2, \mathbb{A}), \quad \hat{e}(\sigma_4, g) \stackrel{?}{=} \hat{e}(\sigma_2, \mathbb{B}), \\ \hat{e}(\sigma_6, g) &\stackrel{?}{=} \hat{e}(H(\Gamma), \mathbb{X}), \quad g^{\sigma_8} \stackrel{?}{=} \sigma_5 \cdot \mathbb{X}^{\sigma_7} \\ \sigma_7 &\stackrel{?}{=} h(M || \Gamma || \sigma_5) + h(\hat{e}(\sigma_3, \prod_{i=1}^l CV_i) \hat{e}(\sigma_4, \prod_{i=1}^l CR_i)), \end{aligned}$$

hold. If it does not hold, then  $V$  outputs **reject**. Otherwise, it outputs **accept**.

## 4.1 Security Analysis

### Unforgeability

**Theorem 1.** *The above multi-level controlled signature scheme is existentially unforgeable under an adaptive chosen message and credential exposure attack if the CDH assumption holds in the random oracle model.*

*Proof.* Assume that there exists a forger algorithm  $\mathcal{A}$  running the existentially unforgeable game defined in Section 3.1. Then we will show that, by using  $\mathcal{A}$ , an adversary  $\mathcal{F}$  solves the CDH problem.

We now begin with the the construction of oracles. To begin with,  $\mathcal{F}$  runs *Setup* and *TKeyGen* to obtain a system parameter **param**, a secret key  $sk_{TA}$  and a public key of  $TA$ . Next, on input  $g, g^x$  and  $g^y$  as an instance of the CDH problem,  $\mathcal{F}$  sets  $\mathbb{X} = g^x; \mathbb{W} = \mathbb{X}^a; \mathbb{U} = \mathbb{X}^b$  as the signer public key  $pk_S$ .  $\mathcal{F}$  sets  $g^y$  as one of the answers for the hash query to the random oracle. Then,  $\mathcal{F}$  construct oracles as follows:

**$\mathcal{HO}$  oracle:** On input a string  $\Gamma$ , if it is a request for a hash value of  $H(\Gamma)$ ,  $\mathcal{HO}$  oracle randomly choose  $d \stackrel{\$}{\leftarrow} \{0, 1\}$  such that the probability of  $d = 1$  is  $\frac{1}{q_H}$ . If  $d = 1$ , set  $H(\Gamma) = g^y$  and return  $H(\Gamma)$ . Otherwise,  $l \stackrel{\$}{\leftarrow} \mathbb{Z}_p; H(\Gamma) = g^l$  and return  $H(\Gamma)$ . In case of  $h(\Gamma)$ ,  $\mathcal{HO}$  chooses  $\iota \stackrel{\$}{\leftarrow} \mathbb{Z}_p$  and then returns  $h(\Gamma) = \iota$ . Then  $\mathcal{HO}$  keeps  $l$  and  $\iota$  in the list and this list can be accessed only by  $\mathcal{F}$ .

**$\mathcal{VCO}$  queries :** On input a secret key  $sk_{TA}$ ,  $\mathcal{VCO}$  runs *CreGen* to generate the credential  $VCR$  for the security level assertion  $A_V = l$  and then returns  $VCR$ .

**$\mathcal{SSO}$  queries :** On input  $MP = "A_V \geq l"$  and a message  $M$ ,  $\mathcal{SSO}$  computes a multi-level controlled signature as follows:

$$\begin{aligned} r, k &\stackrel{\$}{\leftarrow} \mathbb{Z}_p, \quad \sigma_1 = g^r, \quad \sigma_2 = \mathbb{X}^r, \quad \sigma_3 = \mathbb{W}^r, \quad \sigma_4 = \mathbb{U}^r, \\ \Gamma &= \sigma_1 || \sigma_2 || \sigma_3 || \sigma_4 || pk_S || pk_{TA} || MP. \end{aligned}$$

Before processing the next step, on access to the list of  $l$  and  $\iota$ ,  $\mathcal{F}$  checks whether  $H(\Gamma) \stackrel{?}{=} g^y$ . If it holds, output  $\perp$ . Otherwise,  $\mathcal{F}$  gives  $l$  to  $\mathcal{SSO}$ . Next,  $\mathcal{F}$  randomly selects  $\iota' \stackrel{\$}{\leftarrow} \mathbb{Z}_p; K \stackrel{\$}{\leftarrow} \mathbb{G}_1$  and  $\mathcal{F}$  adds  $\iota', h(M || \Gamma || K)$  to the list. Then,  $\mathcal{F}$  returns  $K$  to  $\mathcal{SSO}$ . As a result,  $\mathcal{SSO}$  computes the rest of signature as follows:

$$z \stackrel{\$}{\leftarrow} \mathbb{Z}_p, \quad \sigma_8 = z, \quad \sigma_6 = \mathbb{X}^l, \quad \sigma_7 = h(\hat{e}(\mathbb{X}, W_l)^{\mu \cdot r}) + h(M || \Gamma || K), \quad \sigma_5 = g^{\sigma_8} \mathbb{X}^{-\sigma_7}.$$

At the end of the process, on input  $\sigma_5$  from  $\mathcal{SSO}$ ,  $\mathcal{F}$  updates  $\iota', h(M || \Gamma || \sigma_5)$  to the list. Hence, a multi-level controlled signature on message  $M$  is  $\sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6, \sigma_7, \sigma_8)$ .  $\mathcal{SSO}$  then responds with  $M, \sigma, MP$ .

Now, we begin the game by giving an access to the above oracles to  $\mathcal{A}$ . Assume that  $\mathcal{A}$  always makes a query for a string or a message to  $\mathcal{HO}$  oracle before it outputs a potential forgery, denoted by  $M^*, \sigma^*, MP^*$ . After executing an adaptive strategy with the above oracles,  $\mathcal{A}$  outputs a forgery  $\sigma^*$  on a message  $M^*$  with respect to  $MP^*$ .  $\mathcal{A}$  wins the game if a multi-level controlled signature  $\sigma^*$  on message  $M^*$  with respect to  $MP^*$  is valid and is not an output from the  $\mathcal{SSO}$  queries.

We denote by  $\epsilon$  the success probability  $\text{Succ}_{\text{EUF-MLCS}}^{CM-A}(\cdot)$  that  $\mathcal{A}$  wins the game. Let  $e$  be the base of the natural logarithm. As we mentioned early, a query for a hash of a string or message to  $\mathcal{HO}$  is always issued before  $\mathcal{A}$  issues a query for a signature to  $\mathcal{SSO}$  queries, hence,  $q_H \geq q_S$ . Now, we can analyze the success probability that  $\mathcal{A}$  outputs a signature  $\sigma^*$  on message  $M^*$  with respect to  $MP^*$ , where  $\sigma_6^* = H(\Gamma)^x = (g^y)^x$ , and wins the above game as follows:

- $E_1$ :  $\mathcal{F}$  does not abort during the issuing of queries to the  $\mathcal{SSO}$ . The probability of this event  $\Pr[E_1]$  is  $(1 - \frac{1}{q_H})^{q_S}$ . It is because  $\mathcal{A}$  needs to have at least one query for  $H(\Gamma)$  to output  $\sigma_6^*$  which is a part of forgery. Since  $q_H \geq q_S$ , the upper bound for  $\mathcal{SSO}$  queries is then  $q_H - 1$  and  $\Pr[E_1] \geq (1 - \frac{1}{q_H})^{q_H-1} \approx \frac{q_H}{e \cdot (q_H-1)}$ .
- $E_2$ :  $\mathcal{F}$  does not abort after  $\mathcal{A}$  output  $\sigma^*$ .  $\mathcal{F}$  aborts the experiment after  $\mathcal{A}$  output  $\sigma^*$  when only  $H(\Gamma) \neq g^y$ . Therefore, the probability of this event is greater than  $(1 - \frac{1}{q_H})^{q_H-1} \approx \frac{q_H}{e \cdot (q_H-1)}$ .

To summarize the probability,  $\mathcal{A}$  wins the above game and outputs a signature  $\sigma^*$  on a message  $M^*$ , where  $H(\Gamma) = g^y$  and  $\sigma_6^* = H(\Gamma)^x$ , with a probability equal to  $\Pr[\text{Succ}_{\text{EUF-MLCS}}^{CM-A}] \cdot \Pr[\text{Succ}_{\text{EUF-MLCS}}^{CM-A} | E_1 | E_2] \geq \epsilon (\frac{q_H}{e \cdot (q_H-1)})^2$ . From the above results,  $\mathcal{F}$  outputs  $\sigma_6^* = H(\Gamma)^x = g^{xy}$  as an answer to CDH problem and the above success probability shows that our multi-level controlled signature scheme secures against existentially unforgeable under an adaptive chosen message and credential exposure attack if the success probability of solving CDH problem is negligible.

## Coalition-resistance

**Theorem 2.** *The above multi-level controlled signature scheme is existentially coalition-resistance against the adaptively chosen message and chosen multi-level security policy distinguisher  $\mathcal{A}_{\text{CR-MLCS}}^{CMP-A}$  if the DBDH assumption is hold in the random oracle model.*

Due to the page limitation, please find the proof for Theorem 2 in the full version of this paper [12].

## 5 The Second MLCS Scheme

In this section, we present the second construction of MLCS schemes. The scheme is described as follows.

*Setup* : On input a security parameter  $\mathcal{K}$ , a trusted third party randomly selects a prime  $p \approx \text{poly}(1^{\mathcal{K}})$ . Choose a random generator  $g \in \mathbb{G}_1$  and a bilinear mapping function  $\hat{e}$ . Select two hash functions  $H(\cdot)$  and  $h(\cdot)$ . Let  $\text{param} = (p, \hat{e}, g, H, h)$  denote the system parameters. Then, *Setup* returns  $\text{param}$ .

*TKeyGen* : Let  $n$  be a number of security levels. On input a system parameters  $\text{param}$ , a trusted authority  $TA$  randomly generates a private key  $sk_{TA}$  and a public key  $pk_{TA}$  for each security level as follows: select random integers  $\mu, a, b, w_1, \dots, w_n \in \mathbb{Z}_p$ . Let  $pk_{TA} = (U = g^\mu, \mathbb{A} = g^a, \mathbb{B} = g^b, W_1 = g^{w_1}, \dots, W_n = g^{w_n})$  denote a public key. Then, *TKeyGen* returns  $sk_{TA} = (\mu, a, b, w_1, \dots, w_n)$  as a private key of the trusted authority and  $pk_{TA} = (\mathbb{A}, \mathbb{B}, U, W_1, \dots, W_n)$  as a public key of the trusted authority.

*SKeyGen* : On input a system parameters  $\text{param}$ , a signer  $S$  randomly generates a private key  $sk_S$  and a public key  $pk_S$  as follows. Choose a random integer  $x \in \mathbb{Z}_p$ . Let set  $pk_S = (\mathbb{X} = g^x, \mathbb{U} = U^x, \mathbb{W}_1 = W_1^x, \dots, \mathbb{W}_n = W_n^x)$  to a public key. Then, *SKeyGen* returns  $sk_S = x$  as a private key of the signer and  $pk_S = (\mathbb{X}, \mathbb{U}, \mathbb{W}_1, \dots, \mathbb{W}_n)$  as a public key of the signer.

*CreGen* : Let  $A_V$  indicates a security level of verifier. On input a system parameters **param**, the trusted authority's public key  $pk_{TA}$ , the trusted authority's private key  $sk_{TA}$  and a security level of verifier  $A_V = l$  that verifier is satisfied to obtain, a trusted authority  $TA$  randomly generates a credential strings  $VCR = (CV, CR)$  as follows:  $TA$  randomly selects  $s \in \mathbb{Z}_p^*$  and computes each credential  $CV = g^s$ ;  $CR = g^{((a \cdot b - s \cdot \mu) / w_l)}$  and then returns  $VCR = (CV, CR)$  to the verifier as a credential for a security level assertion  $A_V = l$ . Verifier checks the validity of  $CV, CR$  as follows:

$$\hat{e}(A, B) \stackrel{?}{=} \hat{e}(U, CV) \hat{e}(W_l, CR).$$

*Sign* : Given **param**,  $pk_{TA}, sk_S, pk_S, MP = "A_V \geq l"$  and a message  $M$ ,  $S$  computes a multi-level controlled signature  $\sigma$  on a message  $M$  as follows:

$$\begin{aligned} r, k &\stackrel{\$}{\leftarrow} \mathbb{Z}_p, \sigma_1 = g^r, \sigma_2 = \mathbb{X}^r, \sigma_3 = (\mathbb{W}_l^r, \dots, \mathbb{W}_n^r) \sigma_4 = \mathbb{U}^r, \\ \Gamma &= \sigma_1 || \sigma_2 || \sigma_3 || \sigma_4 || pk_S || pk_{TA} || MP, \\ \sigma_5 &= g^k, \sigma_6 = H(\Gamma)^x, \sigma_7 = h(\hat{e}(A, B)^{x \cdot r}) + h(M || \Gamma || \sigma_5), \sigma_8 = k + \sigma_7 \cdot x. \end{aligned}$$

The multi-level controlled signature on a message  $M$  is  $\sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6, \sigma_7, \sigma_8)$ .  $S$  publishes  $M, \sigma, MP$ .

*Verify* : Let  $VCR = (CV, CR)$  be a credential that verifier possessed for a security level " $A_V = t$ ", where  $l \leq t \leq n$ . Let phrase  $\Gamma = \sigma_1 || \sigma_2 || \sigma_3 || \sigma_4 || pk_S || pk_{TA} || MP$ . Given  $pk_S, pk_{TA}, pk_V, VCR, MP = "A_V \geq l"$ ,  $\sigma$  and a message  $M$ , a verifier  $V$  checks whether, for  $i = l$  to  $n$ ,  $\hat{e}(\sigma_{3,i}, g) \stackrel{?}{=} \hat{e}(\sigma_2, W_i)$  and, then, check whether

$$\begin{aligned} \hat{e}(\sigma_1, \mathbb{X}) &\stackrel{?}{=} \hat{e}(\sigma_2, g), \hat{e}(\sigma_4, g) \stackrel{?}{=} \hat{e}(\sigma_2, U), \hat{e}(\sigma_6, g) \stackrel{?}{=} \hat{e}(H(\Gamma), \mathbb{X}), \\ \sigma_7 &\stackrel{?}{=} h(M || \Gamma || \sigma_5) + h(\hat{e}(\sigma_4, CV) \hat{e}(\sigma_{3,t}, CR)), g^{\sigma_8} \stackrel{?}{=} \sigma_5 \cdot \mathbb{X}^{\sigma_7} \end{aligned}$$

hold. If it does not hold, then  $V$  outputs **reject**. Otherwise, it outputs **accept**.

## 5.1 Security Analysis

### Unforgeability

**Theorem 3.** *Our second multi-level controlled signature scheme is existentially unforgeable under an adaptive chosen message and credential exposure attack if the CDH assumption holds in the random oracle model.*

*Proof.* Assume that there exists a forger algorithm  $\mathcal{A}$  running the existentially unforgeability game defined in Section 3.1. Then we will show that, by using  $\mathcal{A}$ , an adversary  $\mathcal{F}$  solves the CDH problem.

We now begin with the the construction of oracles. To begin with,  $\mathcal{F}$  runs *Setup* and *TKeyGen* to obtain a system parameter **param**, a secret key  $sk_{TA}$  and a public key of  $TA$ . Next, on input  $g, g^x$  and  $g^y$  as an instance of the CDH problem,  $\mathcal{F}$  sets  $\mathbb{X} = g^x; \mathbb{U} = \mathbb{X}^\mu; \mathbb{W}_1 = \mathbb{X}^{w_1}; \dots; \mathbb{W}_n = \mathbb{X}^{w_n}$  as the signer public key  $pk_S$ .  $\mathcal{F}$  sets  $g^y$  as one of the answers for the hash query to the random oracle. Then,  $\mathcal{F}$  constructs oracles as follows:

**$\mathcal{HO}$  oracle:** On input a string  $\Gamma$ , if it is a request for a hash value of  $H(\Gamma)$ ,  $\mathcal{HO}$  oracle randomly choose  $d \stackrel{\$}{\leftarrow} \{0, 1\}$  such that the probability of  $d = 1$  is  $\frac{1}{q_H}$ . If  $d = 1$ , set  $H(\Gamma) = g^y$  and return  $H(\Gamma)$ . Otherwise,  $l \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ ;  $H(\Gamma) = g^l$  and return  $H(\Gamma)$ . In case of  $h(\Gamma)$ ,  $\mathcal{HO}$  chooses  $\iota \stackrel{\$}{\leftarrow} \mathbb{Z}_p$  and then returns  $h(\Gamma) = \iota$ . Then  $\mathcal{HO}$  keeps  $l$  and  $\iota$  in the list and this list can be accessed only by  $\mathcal{F}$ .

**$\mathcal{VCO}$  queries :** On input a secret key  $sk_{TA}$ ,  $\mathcal{VCO}$  runs *CreGen* to generate the credential  $VCR$  for the security level assertion  $A_V = l$  and then returns  $VCR$ .

**SSO queries :** On input  $MP = “A_V \geq l”$  and a message  $M$ ,  $SSO$  computes a multi-level controlled signature as follows:

$$r, k \xleftarrow{\$} \mathbb{Z}_p, \sigma_1 = g^r, \sigma_2 = \mathbb{X}^r, \sigma_3 = (\mathbb{W}_l^r, \dots, \mathbb{W}_n^r), \sigma_4 = \mathbb{U}^r, \\ \Gamma = \sigma_1 || \sigma_2 || \sigma_3 || \sigma_4 || pk_S || pk_{TA} || MP.$$

Before processing the next step, on access to the list of  $l$  and  $\iota$ ,  $\mathcal{F}$  checks whether  $H(\Gamma) \stackrel{?}{=} g^y$ . If hold, output  $\perp$ . Otherwise,  $\mathcal{F}$  gives  $l$  to  $SSO$ . Next,  $\mathcal{F}$  randomly selects  $\iota' \xleftarrow{\$} \mathbb{Z}_p$ ;  $K \xleftarrow{\$} \mathbb{G}_1$  and  $\mathcal{F}$  adds  $\iota', h(M || \Gamma || K)$  to the list. Then,  $\mathcal{F}$  returns  $K$  to  $SSO$ . As a result,  $SSO$  computes the rest of signature as follows:

$$z \xleftarrow{\$} \mathbb{Z}_p, \sigma_8 = z, \sigma_6 = \mathbb{X}^l, \sigma_7 = h(\hat{e}(\mathbb{X}, \mathbb{A})^{b \cdot r}) + h(M || \Gamma || K), \sigma_5 = g^{\sigma_8} \mathbb{X}^{-\sigma_7}.$$

In the end of the process, on input  $\sigma_5$  from  $SSO$ ,  $\mathcal{F}$  updates  $\iota', h(M || \Gamma || \sigma_5)$  to the list. Hence, a multi-level controlled signature on message  $M$  is  $\sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6, \sigma_7, \sigma_8)$ .  $SSO$  then responds with  $M, \sigma, MP$ .

Now, we begin the game by giving an access to the above oracles to  $\mathcal{A}$ . Assume that  $\mathcal{A}$  always make a query for a string or a message to  $\mathcal{HO}$  oracle before it outputs a potential forgery, denoted by  $M^*, \sigma^*, MP^*$ . After executing an adaptive strategy with the above oracles,  $\mathcal{A}$  outputs a forgery  $\sigma^*$  on a message  $M^*$  with respect to  $MP^*$ .  $\mathcal{A}$  wins the game if a multi-level controlled signature  $\sigma^*$  on message  $M^*$  with respect to  $MP^*$  is valid and is not an output from the  $SSO$  queries.

We denote by  $\epsilon$  the success probability  $Succ_{EUF-MLCS}^{CM-A}(\cdot)$  that  $\mathcal{A}$  wins the game. Let  $e$  be the base of the natural logarithm. As we mentioned early, a query for a hash of a string or message to  $\mathcal{HO}$  is always issued before  $\mathcal{A}$  issues a query for a signature to  $SSO$  queries, hence,  $q_H \geq q_S$ . Now, we can analyze the success probability that  $\mathcal{A}$  outputs a signature  $\sigma^*$  on message  $M^*$  with respect to  $MP^*$ , where  $\sigma_6^* = H(\Gamma)^x = (g^y)^x$ , and wins the above game as follows:

- $E_1$ :  $\mathcal{F}$  does not abort during the issuing of queries to the  $SSO$ . The probability of this event  $\Pr[E_1]$  is  $(1 - \frac{1}{q_H})^{q_S}$ . It is because  $\mathcal{A}$  needs to have at least one query for  $H(\Gamma)$  to output  $\sigma_6^*$  which is a part of forgery. Since  $q_H \geq q_S$ , the upper bound for  $SSO$  queries is then  $q_H - 1$  and  $\Pr[E_1] \geq (1 - \frac{1}{q_H})^{q_H - 1} \approx \frac{q_H}{e \cdot (q_H - 1)}$ .
- $E_2$ :  $\mathcal{F}$  does not abort after  $\mathcal{A}$  output  $\sigma^*$ .  $\mathcal{F}$  aborts the experiment after  $\mathcal{A}$  output  $\sigma^*$  when only  $H(\Gamma) \neq g^y$ . Therefore, the probability of this event is greater than  $(1 - \frac{1}{q_H})^{q_H - 1} \approx \frac{q_H}{e \cdot (q_H - 1)}$ .

To summarize the probability,  $\mathcal{A}$  wins the above game and outputs a signature  $\sigma^*$  on a message  $M^*$ , where  $H(\Gamma) = g^y$  and  $\sigma_6^* = H(\Gamma)^x$ , with a probability equal to  $\Pr[Succ_{EUF-MLCS}^{CM-A}] \cdot \Pr[Succ_{EUF-MLCS}^{CM-A} | E_1 | E_2] \geq \epsilon (\frac{q_H}{e \cdot (q_H - 1)})^2$ . From the above results,  $\mathcal{F}$  outputs  $\sigma_6^* = H(\Gamma)^x = g^{xy}$  as an answer to CDH problem and the above success probability shows that our multi-level controlled signature scheme secures against existentially unforgeable under an adaptive chosen message and credential exposure attack if the success probability of solving CDH problem is negligible.

## Coalition-resistance

**Theorem 4.** *Our second multi-level controlled signature scheme is existentially coalition-resistance against the adaptively chosen message and chosen multi-level security policy distinguisher  $\mathcal{A}_{CR-MLCS}^{CMP-A}$  if the DBDH assumption is hold in the random oracle model.*

Due to the page limitation, please find the proof for Theorem 4 in the full version of this paper [12].

## 6 Conclusion

We presented a security model for MLCS schemes to capture the message integrity and authenticity, together with authorization of the verifiers. Two concrete schemes and their proof of security

have been presented. A signature of the first scheme has a constant size. A private information (credentials) of verifier is shorter in the second scheme. Our schemes are shown to be secure in our model. The comparison between the two schemes is provided in Fig 1. It is interesting to study how to provide a constant size signature while the size of verifier’s credential is also constant. Moreover, it is also interesting to study how to construct MLCS schemes in the standard model. We discussed the insecure generic construction from HIBE, however, it is also interesting to study how to construct a generic construction scheme that will be secure in our model such that the verification can be done only by an appropriate verifier. Note that, in Fig 1,  $l$  is a security level in the multi-level security policy where  $MP = “A_V \geq l”$ .  $n$  is the number of security level. Let  $E$  denote a computation of exponential in  $\mathbb{G}_1$  or  $\mathbb{G}_T$ . Let  $M$  be a computation of multiplication in  $\mathbb{G}_1$ . Let  $P$  be a computation of bilinear pairing function  $\hat{e}$ .

Version / Size of	First Scheme	Second Scheme
$PK_{TA}$	$(2n + 2) \mathbb{G}_1 $	$(n + 3) \mathbb{G}_1 $
$SK_{TA}$	$(3n + 2) p $	$(n + 3) p $
$PK_S$	$3 \mathbb{G}_1 $	$(n + 1) \mathbb{G}_1 $
$SK_S$	$ p $	$ p $
$VCR_V$	$(2l) \mathbb{G}_1 $	$2 \mathbb{G}_1 $
Signature	$6 \mathbb{G}_1  + 2 p $	$(5 + n - l) \mathbb{G}_1  + 2 p $
Signing Computation	$7E + M + P$	$(6 + n - l)E + M + P$
Verification Computation	$E + 2lM + 10P$	$E + (2(n - l) + 8)P$

**Fig. 1.** The comparison of two concrete schemes.

## References

1. W. Bagga and R. Molva. Policy-based cryptography and applications. In A. S. Patrick and M. Yung, editors, *Financial Cryptography*, volume 3570 of *Lecture Notes in Computer Science*, pages 72–87. Springer, 2005.
2. D. Boneh. The decision Diffie-Hellman problem. In J. Buhler, editor, *ANTS*, volume 1423 of *Lecture Notes in Computer Science*, pages 48–63. Springer, 1998.
3. D. Boneh, X. Boyen, and E.-J. Goh. Hierarchical Identity Based encryption with constant size ciphertext. In R. Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456. Springer, 2005.
4. D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. In J. Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.
5. X. Boyen and B. Waters. Anonymous hierarchical identity-based encryption (without random oracles). In C. Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 290–307. Springer, 2006.
6. S. S. M. Chow, L. C. K. Hui, S.-M. Yiu, and K. P. Chow. Secure hierarchical Identity Based signature and its application. In J. Lopez, S. Qing, and E. Okamoto, editors, *ICICS*, volume 3269 of *Lecture Notes in Computer Science*, pages 480–494. Springer, 2004.
7. W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.
8. C. Gentry and A. Silverberg. Hierarchical ID-based cryptography. In Y. Zheng, editor, *ASIACRYPT*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer, 2002.
9. J. Horwitz and B. Lynn. Toward hierarchical identity-based encryption. In L. R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 466–481. Springer, 2002.
10. M. Jakobsson, K. Sako, and R. Impagliazzo. Designated Verifier Proofs and Their Applications. *Advances in Cryptology - Eurocrypt '96, Lecture Notes in Computer Science 1070*, pages 143 – 154, 1996.
11. A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53, 1984.
12. P. Thorncharoensri, W. Susilo, and Y. Mu. Multi-level controlled signature (full version). *can be obtained from the first author*, 2012.