# A Certificateless Ordered Sequential Aggregate Signature Scheme Secure against Super Adversaries

Naoto Yanai*
*Graduate School of Systems and Information Engineering*
*University of Tsukuba*
*Tsukuba, Japan*
yanai@cipher.risk.tsukuba.ac.jp

Raylin Tso
*Department of Computer Science*
*National Chengchi University*
*Taipei, Taiwan*
raylin@cs.nccu.edu.tw

Masahiro Mambo
*Institute of Science and Engineering*
*Kanazawa University*
*Kanazawa, Japan*
mambo@ec.t.kanazawa-u.ac.jp

Eiji Okamoto,
*Graduate School of Systems and Information Engineering*
*University of Tsukuba*
*Tsukuba, Japan*
okamoto@risk.tsukuba.ac.jp

**Abstract**

Certificateless cryptosystem is a hybrid scheme of traditional PKI and ID-based scheme and has positive aspects of both of PKI and ID-based cryptosystem, i.e. solving key escrow problem and certificate management problem simultaneously. Cryptographic schemes constructed in such a hybrid setting, generally called certificateless setting, retain these positive aspects and have been extensively studied recently. To the best of our knowledge, an ordered sequential aggregate signature (OSAS) scheme, which is a signature scheme verifying both the validity of a document and a signing order of a group of signers, has never been proposed in the certificateless setting. Therefore we propose an OSAS scheme in a certificateless setting called certificateless ordered sequential aggregate signature (CLOSAS) scheme. Our proposed scheme has advantages in its communication cost and the security proof. In particular, its signature size is fixed with respect to the number of signers, and the security is proven in the random oracle model against super adversaries that are the strongest adversary in certificateless signature scheme. Our scheme resists KGC's malicious activities associated with key escrow and forgery of signatures as long as both of each user and KGC involve directly in a key generation.

**Keywords**: Key escrow problem, certificateless setting, ordered sequential aggregate signature scheme, super adversary, random oracle model, full aggregation

## 1 Introduction

### 1.1 Motivation

One of the main problems in public key cryptosystem is to guarantee a relation between a user and its own public key. In general, a public key in traditional public key cryptography such as RSA encryption[1] is a random value, and we need a method to bind the user with the public key. A general way to solve this problem is to utilize a public key infrastructure (PKI) in which a trusted third party called certification authority (CA) issues a certificate to bind the user with the public key. However, in the PKI the

*Corresponding author: SB822, Third Area, University of Tsukuba, Tennoudai 1-1-1, Tsukuba-city, Ibaraki, Japan, Phone: +81-029-853-5600 (ext: 8208) management cost for certificates is expensive in that it involves certificate revocation, distribution and verification of public keys. This problem in the PKI is called *certificate management problem*.

As an approach to overcome the problem in the PKI, identity-based (ID-based) cryptosystem[2] has been studied in recent year. In an ID-based cryptosystem, each user has ID information such as an e-mail address and can use the ID as his/her own public key. In general, ID is unique information for each user and is publicly known. In contrast to PKI, users in ID-based cryptosystem do not need a certificate to relate a user to his/her public key, ID. However, ID-based cryptosystem has an inherent problem, called *key escrow problem*, in which a key generation center (KGC) knows secret keys for all users in the system. This problem occurs because secret keys of all the users are computed from KGC's master secret key and users' ID. This implies that the KGC must be trusted in ID-based cryptosystem. In other words, malicious KGC's can easily read contents of encrypted communications and ID-based systems intrinsically contain such an insider's threat. In fact, users cannot always trust KGC's since malicious KGC's who does not honestly run the algorithm exist[3].

In order to overcome this problem, Al-Riyami et al. proposed the *certificateless cryptosystem*[4] which is a hybrid cryptosystem of PKI and ID-based cryptosystem. In the certificateless cryptosystem, the key of each user consists of a pair of secret key and public key depending upon both PKI and ID-based cryptosystem. In particular, after given the secret value in ID-based cryptosystem called partial private key, each user generates a secret value which is a random number in PKI. Then the user sets the secret value and the partial private key as a full secret key of him/her, and a value computed from the secret value in PKI and his/her ID as a corresponding public key. [1] A sender/verifier uses the public key for the encryption/verification of data, and a receiver/signer uses the full secret key for the decryption/signing of data. The certificateless cryptosystem has positive aspects of both PKI and ID-based system. In particular, the confidentiality or the validity of the data of users are guaranteed even if the KGC is malicious, because the KGC does not know the secret value generated by the user in PKI. In addition, the user can implicitly confirm an owner of the public key without the certificate since the user needs ID as the part of the public key to encrypt/verify the data. Therefore, constructing certificateless cryptographic schemes such as signature schemes is a meaningful work.

As one of main applications in cryptography, digital signature scheme which guarantees the validity of an electronic document is a famous tool and has been studied by many researchers[5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37]. Multisignature scheme by Itakura et al.[14] is suitable for a situation in which the validity of the document should be guaranteed by all associated persons, and its communication cost has an advantage in which the data size of a signature is smaller than an individual signing which is just to collect signatures of all the associated signers. After many multisignature schemes were proposed[8, 22, 25, 28], Boneh et al. proposed *aggregate signature (AS) scheme* as a generalized scheme of multisignature scheme in 2003[6]. Each signer in an AS scheme can sign an individual document, and hence aggregate signature scheme has been focused as one of main topics in digital signature scheme in recent work[5, 9, 10, 11, 12, 18, 20, 21, 24, 26, 27, 29, 31, 35].

Among them, *ordered sequential aggregate signature (OSAS) scheme* by Lysyanskaya et al.[20] is an AS scheme which verifies both the validity of the document and a signing order and, as described in section 1.4, is adopted to some application such as *secure-border gateway protocol (S-BGP)*[38]. However, to the best of our knowledge, the existing OSAS schemes have been proposed in either PKI or ID-based scheme, and such a scheme in a certificateless setting has never been proposed. Certificateless cryptosystem is the advanced scheme in contrast to PKI and ID-based scheme as described above, and

[1]Several researchers avoid to view ID as public key in ID-based system since ID is not a randomly generated value as set in the traditional PKI. Even so, ID-based system can be judged as an answer to the question, "Is it possible to construct a public-key system with a fixed-value public key?" and we describe ID as a part of the public key.

thus in this paper we propose an OSAS scheme in the certificateless setting, i.e. *certificateless ordered sequential aggregate signature (CLOSAS) scheme*.

In addition to constructing the first CLOSAS scheme, we also discuss the security against the super adversaries[39] who are the strongest adversary known in the certificateless cryptosystem. As described in section 1.2.5, the super adversaries can implicitly access a black box knowledge extractor which extracts a secret key from a corresponding public key without being detected by a target signer, and several schemes[40, 12] secure against non-super adversaries become insecure against the super adversary. To avoid any unexpected security degradation, one should guarantee the highest security level and proving the security against the super adversaries is meaningful work. Note that we do not discuss a malicious activity such that KGC impersonates target users by generating pairs of full secret key and public key. To the author's knowledge, no existing certificateless scheme prevents this type of KGC's impersonation and it still remains open to construct CLOSAS scheme guaranteeing the validity of both of documents and the signing order even under the KGC's impersonation. In our discussion framework, the validity of both of documents and the signing order is guaranteed as long as either the secret values in PKI or the partial private keys are kept secret. Further note that this paper is an extended version of the paper [33]. While the super adversaries were not considered in the previous work, we discuss security against the super adversaries in this paper. We show a rigorous proof that the proposed scheme is secure against the super adversary in the random oracle model if and only if solving CDH problem is difficult.

## 1.2 Achievement for Our Construction

Our newly proposed scheme has the following features.

### 1.2.1 Certificateless Property

Certificateless cryptosystem does not need a certificate generated by CA to verify a user's public key, so it does not have the certificate management problem suffered in traditional PKI-based public key cryptosystem. On the other hand, it also solves the key escrow problem suffered in ID-based cryptosystems since a secret key generated by each user in PKI is an unknown value for a malicious KGC. Therefore, we propose our OSAS scheme in the certificateless setting.

In a security notion in public key cryptosystems, Girault[41] defined three security levels for a trusted authority as follows:

level-1 KGC knows a secret key for any user, and can impersonate the user with the secret key without being detected.

level-2 KGC does not know a secret key for any user, but can impersonate the user with the secret key without being detected by generating a fake secret key.

level-3 KGC does not know a secret key for any user. In addition, KGC cannot impersonate the user with the secret key even by generating a fake secret key since its impersonation can be detected.

The security model used for the analysis of our scheme does not capture an actively malicious KGC who generates a pair of a secret key and its corresponding public key for any user. Namely, our proposed scheme achieves level-2 security. However, based on the idea of [30] proposed by Wu et. al. in 2009, it is easy to modify our certificateless signature into a new kind of signature scheme named *certificate-based signature scheme* [42, 43, 30] in which the Girault's Level-3 security can be achieved. But, with this modification, the public key $PK_{ID}$ of an entity $ID$ will not be able to update at any time without any assistance from KGC whereas this is possible in our scheme. Therefore, here we only discuss how to protect a certificateless signature scheme under the assumption that a secret value in either PKI or

ID-based cryptosystem is kept secret and that malicious activities of KGC are restricted not to fake a pair of secret and public keys described above, i.e. security level-2. In other words, our scheme can resist signature forgery unless KGC impersonates a target signer by generating a key of the target signer.

### 1.2.2 Ordered Sequential Aggregate Signature Scheme

According to Selvi et al.[26], three types exist as AS scheme, i.e. *general aggregate signature (GAS) scheme*, *sequential aggregate signature (SAS) scheme* and ordered sequential aggregate signature (OSAS) scheme. GAS scheme is an aggregate signature scheme that each signer's signature is generated in parallel, and then aggregates these signatures into one signature with an interactive process. On the other hand, SAS scheme and OSAS scheme have no aggregate phase described in [6] and its signature is generated by executing both signing and aggregation for each signer in turn. However, while the signing order in SAS scheme has no meaning, the signing order can be verified in OSAS scheme. Our proposed scheme is the OSAS scheme.

### 1.2.3 Full Aggregation

For signature size, Selvi et al. described the notion of *full aggregation* and *partial aggregation* in [27]. The former means that the signature size in the scheme is fixed with respect to the number of signers, and the latter means the signature is linear. Hence, achieving the full aggregation means an efficient scheme for the communication cost. Our proposed scheme achieves the full aggregation.

### 1.2.4 Order Flexibility

Mitomi et al. described an order flexibility in [22]. This notion is intuitively that the signing order should not be included in public information. Achieving this property means that the signers can easily change the signing order. Our proposed scheme achieves this property.

### 1.2.5 Security against Super Adversary

Since public keys in certificateless cryptosystems are not certified by certificates, these public keys can be replaced by an adversary[39, 19]. According to Huang et al.[39], there are three types of the adversary, normal, strong and super. The normal adversary cannot obtain signatures of a target signer once he/she replaces the public key of the target signer. The strong adversary can obtain signatures of the target signer by providing a secret value corresponding to the replaced public key for a challenger in the security model described in section 3.2. The super adversary can also obtain signatures of the target signer but without providing the secret value for the challenger. During the attack, the super adversary can replace a public key $pk_A$ of a target signer Alice with a public key $pk_B$ of another target signer Bob while such an attack cannot be performed by the strong adversary that cannot compute a secret key corresponding to $pk_B$. This means that the super adversary can access Alice as a black box knowledge extractor for the secret value of Bob without being detected by Bob, because in this scenario signatures, which are output of Alice, are computed from the secret value of Bob. Namely, the adversary trying to forge a signature of the target signer can obtain secret-key related information without being detected by the target signer. In this sense, the super adversary can be judged as the strongest adversary among three types of adversaries.

### 1.2.6 Rigorous Proof in Random Oracle Model

In a security proof, we adopt to prove in the random oracle model[44]. In general the construction in the standard model is more rigorous than security analysis in the random oracle model([45]), but it is also true that the construction in the random oracle model is more efficient than those in the standard model. Therefore, we prove the security in the random oracle model.

## 1.3 Contribution

In addition to propose the first CLOSAS scheme, our contribution is to prove the security against the super adversaries in CLOSAS scheme. The security model discussed in this paper is a newly formalized security model by applying the notion of the super adversary in [39] to the security model for OSAS scheme in [20]. As described in more detail in section 3.2, this model captures cryptographic insider threats that dishonest users in the signing group collude with malicious KGC's which know partial private keys. Although the existing security models in the certificateless setting represent malicious entities, their models do not capture a security requirement in OSAS scheme, which is the validity of documents and the signing order, because the existing schemes are not CLOSAS scheme. In contrast, the model in this paper is an advanced model that guarantees the validity of both messages and the signing order even if the malicious entities exist.

## 1.4 Application

We sketch an example of applications using CLOSAS scheme. S-BGP which is one of the application as described in section 1.1 is a routing protocol to overcome a vulnerability in border gateway protocol (BGP)[46]. BGP is a routing protocol that establishes Internet traffic between *autonomous systems (ASes)*, but has no guarantee about the validity of the path information. To overcome this problem, S-BGP enforces ASes to send the data via only the authorized AS path. In particular, ASes generate a digital signature to guarantee a relation of each autonomous system and its IP prefix, and the S-BGP router generates a digital signature to guarantee a neighbor AS. Several papers such as [5, 20] have pointed out that OSAS scheme is suitable tool for S-BGP in which it allows ASes to verify and then forward a propagated data via the authenticated path.

In addition to the advantage described above, we can obtain another advantage by implementing CLOSAS scheme in S-BGP. Main problems for an implementation of S-BGP are a storage of routers and its traffic of the data[47]. In particular, when routers send the data packets in S-BGP, they require to share the public key certificates in advance to verify the signature. Since the packet space is a limited size, attending the certificates with the data to be signed is difficult. In addition, each router also requires large amount of memory to store the certificates and the digital signature sent in S-BGP. Here, we note that the certificateless cryptosystem does not need the public key certificate since the user's ID is bound with its own public key. S-BGP with CLOSAS scheme requires neither sharing the certificates in advance nor large amounts of memory to store the certificates.

S-BGP with CLOSAS scheme is also elegant in the sense of the security against insider threats in contrast to ID-based OSAS scheme. Although S-BGP becomes faster by utilizing ID-based OSAS scheme, this system has some vulnerability in terms of insider threats. In particular, as described in section 1.1 since all the secret keys in ID-based scheme are given by KGC, an adversary such as a malicious KGC who knows a master secret key is able to generate signatures for all signers. This means that authorization of AS path in S-BGP with ID-based OSAS are no longer effective. On the other hand, thanks to the property of the certificateless setting S-BGP with CLOSAS scheme is resistant to the malicious KGC.

### 1.5 Paper Construction

The rest parts of this paper consist as follows. We describe some knowledges to understand this paper in section 2, a general construction of CLOSAS scheme and its security model discussed in this paper in section 3. In section 4 we propose our CLOSAS scheme, and in section 5 we prove the security of the proposed scheme and show an example of more extending application. We evaluate the performance of the proposed scheme in section 6, and conclude about the scheme in section 7.

## 2 Preliminaries

In this section, we introduce some knowledges which are necessary for understanding our paper.

### 2.1 Notations

Let the number of signers be $n$. We denote by $ID_i$ the $i$-th signer if the notation does not cause any confusion. We also denote by $m_i$ a message to be signed by a signer with identity $ID_i$, by $\sigma_i$ the signature generated by $ID_i$, by $msk$ a master secret key, by $mpk$ a master public key, by $sk_i$ a secret key of $ID_i$ and by $pk_i$ its corresponding public key. We define $\psi_i := ID_1 \parallel \cdots \parallel ID_i$ as the signing order from the first signer to $i$-th signer for a group of signers. Let $a \parallel b$ be a concatenation of $a$ and $b$ for all $a, b$, where the concatenation can be easily divided into original elements $a$ and $b$. For simplicity, we denote by $L_i := m_1 \parallel ID_1 \parallel \cdots \parallel m_i \parallel ID_i$ an information including both messages and its signing order to verify the signature.

### 2.2 Bilinear Maps

Our scheme uses bilinear maps. Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of the same prime order $p$. We assume that the Discrete-Logarithm Problem (DLP) in both $\mathbb{G}$ and $\mathbb{G}_T$ are hard.

**Definition 1** (pairing)**.** A pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a maps such that the following conditions hold:

- Bilinearity : For all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}$, $e(u^a, v^b) = e(u, v)^{ab}$.
- Non-degeneracy : For any generator $g \in \mathbb{G}$, $e(g, g) \neq 1_{\mathbb{G}_T}$.
- Computable : There is an efficient algorithm to compute $e(u, v)$ for any $u, v \in \mathbb{G}$.

Through this paper, we denote by $(p, \mathbb{G}, \mathbb{G}_T, e)$ parameters holding the above conditions as a paring parameter.

### 2.3 Security Assumption

In this paper, we use computational Diffie-Hellman (CDH) assumption. CDH assumption is defined as follows.

**Definition 2** (CDH problem)**.** Given $(g, g^a, g^b)$ for all $a, b \in \mathbb{Z}_p$ as input, compute $g^{ab} \in \mathbb{G}$.

**Definition 3** ($(t, \varepsilon)$-CDH assumption)**.** there is no adversary who, given $(g, g^a, g^b)$ for all $a, b \in \mathbb{Z}_p$ as input, can output $g^{ab}$ with probability greater than $\varepsilon$ within the execution time $t$.

### 2.4 Related Work

As described in the previous section, many ordered sequential aggregate signature schemes, which include multisignature schemes, have been proposed so far[5, 7, 8, 9, 10, 13, 15, 16, 17, 18, 20, 22, 23, 24, 25, 27, 28, 29, 31, 32, 34, 36, 37]. Although CLOSAS has never been proposed, we sketch the existing OSAS schemes in PKI and ID-based scheme.

The schemes achieving the full aggregation are in [5, 8, 10, 16, 17, 18, 20, 24, 27, 31, 34] and [36]. However, the schemes in [8, 16, 17, 20, 24, 34] and [36] has no order flexibility, and the scheme in [18] is based on M-LRSW problem[5] which has been shown to be false in [48]. In addition, due to the our analysis based on the claim in the Remark 3 of [11], the scheme in [27] seems to be insecure in that an adversary obtaining multiple signatures in the scheme [27] may be able to recover its corresponding secret key by solving simultaneous equations obtained from these signatures. Hence, we compare the performance of our scheme with rest of the papers [5, 10, 31] in section 6.

On the other hand, several certificateless aggregate signature schemes have been proposed so far[40, 12, 49, 35]. However, the security of the schemes in [40, 12] have never been proven against the super adversary. Although Xiong et al. alleged that their scheme is secure against the super adversary, unfortunately, similarly as the paper [27], it seems that the proof is wrong in the sense that the super adversary may be able to recover the signer's secret key from the reason described on Remark 3 in paper [11]. Hence, to the best of our knowledge, the scheme secure against the super adversary is only the scheme in [35]. Here, the scheme is different from our proposed scheme in that the scheme is not OSAS scheme. In section 6, we also compare the performance of our scheme with the scheme in [35] as the existing certificateless aggregate signature scheme.

## 3 Certificateless Ordered Sequential Aggregate Signature Scheme

### 3.1 General Construction

A CLOSAS scheme consists of following six algorithms. As described above, ordered sequential aggregate signature scheme has no aggregate phase to aggregate signatures, in that the signature is implicitly aggregated in *Signing* phase by each signer.

**Setup** This algorithm is run by KGC. Given a security parameter $1^k$ as input, generate a public parameter *param*, a master secret key *msk* and its corresponding public key *mpk*. Output *param*, *mpk* and *msk*.

**Partial-Private-Key-Extract** This algorithm is run by KGC. Given $param, msk$ and an identity $ID_i$ as input, generate a partial private key $d_i$. Output $d_i$.

**User-Key-Gen** This algorithm is run by each user. Given *param* and his/her identity $ID_i$ as input, generate a secret key $x_i$ and its corresponding public key $y_i$. output $x_i$ and $y_i$.

**Set-Key** This algorithm is run by each user. Given $ID_i, d_i, x_i$ and $y_i$, set $d_i$ and $x_i$ as a full secret key $sk_i$, and $(ID_i, y_i)$ as a corresponding public key $pk_i$. Output $sk_i$ and $pk_i$.

**Signing** This algorithm is run by each user in turn. Given $param, ID_i, sk_i$, $\{m_j\}_{j=1,\cdots,i-1}, \{ID_j\}_{j=1,\cdots,i-1}$, $\psi_{i-1}, \sigma_{i-1}$ and possibly some state information $s$ which is one-time information such as time-stamp as input, check that $\sigma_{i-1}$ is a valid signature on $\{m_j\}_{j=1,\cdots,i-1}$ in $\psi_{i-1}$ by using *Verification* algorithm described below. If not, abort the process. Otherwise, set $\psi_i = \psi_{i-1} \parallel ID_i$. Compute a signature $\sigma_i$ on $m_i$ in $\psi_i$ with $\sigma_{i-1}$ and $s$, then output $\sigma_i$ and $s$.

**Verification** This algorithm is run by a verifier $V$. Given $param, mpk$, $\{ID_j\}_{j=1,\cdots,i}$, $\{pk_j\}_{j=1,\cdots,i}$, $\{m_j\}_{j=1,\cdots,i}, \sigma_i$ and $s$ as input, check that $\sigma_i$ is a valid signature on $\{m_j\}_{j=1,\cdots,i}$ in $\psi_i$. If not, output $reject$. Otherwise, output $accept$.

## 3.2 Security Model

In this section, we define a security model in this paper. Our security model is constructed by applying a notion of super-adversary in [39] to the security model for sequential aggregate signature scheme in [20].

For certificateless signature scheme, we have to discuss two following types of adversaries with different ability. In the security games in this paper, a challenger $\mathscr{C}$ and each adversary who can access a random oracle exist as entities.

**Type 1** This type of adversary, $\mathscr{A}_1$, is a dishonest user who does not have the master secret key *msk* but can replace a public key $y_i$ of any user $ID_i$ with a value chosen by him/her.

**Type 2** This type of adversary, $\mathscr{A}_2$, is a malicious KGC who has *msk* but cannot replace a public key of a target signer.

### 3.2.1 Definition of Oracles

In the security game in this paper, we define the following oracles. We denote by $x^{(j)}$ $j$-th query to access the oracles for all $x$. Here, $\mathscr{C}$ has a certificate list $\mathscr{L}$ to register users' informations.

**Create-User** Given an identity $ID_i$, if $ID_i$ has already been queried, nothing will be output. Otherwise, run the algorithms *Partial-Private-Key-Extract* and *User-Key-Gen*, and generate a partial private key$d_i$, a secret key $x_i$ and a corresponding public key $y_i$. Register $(ID_i, y_i)$ in $\mathscr{L}$ and output $y_i$. In this case, we say that $ID_i$ is created.

**Public-Key-Replace** Given $ID_i$ and $y_i'$ chosen by an adversary, if $ID_i$ has already been created, the original public key for $ID_i$ is replaced with $y_i'$ and re-register $(ID_i, y_i')$ in $\mathscr{L}$. Otherwise, nothing will be output.

**Secret-Value-Extract** Given $ID_i$, if $ID_i$ has already been created, output a secret value $x_i$ corresponding to an original public key $y_i$. Otherwise, nothing will be output. This oracle does not output the secret value corresponding to the replaced public key $y_i'$.

**Partial-Private-Key-Extract** Given $ID_i$, if $ID_i$ has already been created, output a partial private value $d_i$ for $ID_i$. Otherwise, nothing will be output.

**Sign** Given $ID_i, \{m_j\}_{j=1,\cdots,i}, \sigma_{i-1}, \psi_i$, $s$ and a public key $y_i$ of $ID_i$, if $ID_i$ has already been created, output a valid signature $\sigma_i$ on $\{m_j\}_{j=1,\cdots,i}$ in $\psi_i$. Otherwise, nothing will be output. Here $y_i$ may be either the original public key generated by $ID_i$ or a public key replaced by the adversary [2].

### 3.2.2 Game 1

This game is executed between $\mathscr{C}$ and $\mathscr{A}_1$.

**Setup** $\mathscr{C}$ runs the setup algorithm described in the previous section to obtain $param, msk$ and $mpk$. $\mathscr{C}$ gives $param$ and $mpk$ to $\mathscr{A}_1$ but keeps $msk$ to be secret.

**Queries** $\mathscr{A}_1$ can access all the oracles described in section 3.2.1 and obtains the outputs from $\mathscr{C}$.

**Forgery** $\mathscr{A}_1$ outputs a forgery $(\{ID_j^*\}_{j=1,\cdots,n}$, $\{m_j^*\}_{j=1,\cdots,n}$, $\psi_n^*, \sigma_n^*)$ and checks that the following conditions hold.

- $\sigma_n^*$ is a valid signature on $\{m_j^*\}_{j=1,\cdots,n}$ in $\psi_n^*$ under $\{pk_j^*\}_{j=1,\cdots,n}$.
- Exactly one $ID_{i^*}^*$ who has never been queried for partial-private-key-extract oracle exists.
- Each $ID_i^*$ in $\{ID_j^*\}_{j=1,\cdots,n}$ does not appear more than once in $\psi_n^*$.
- For $ID_{i^*}^*$, $m_{i^*}^* \notin \{m_{i^*}^{(1)}, \cdots, m_{i^*}^{(q_s)}\}$ or $\psi_{i^*}^* \notin \{\psi_{i^*}^{(1)}, \cdots, \psi_{i^*}^{(q_s)}\}$ holds, where $q_s$ will be defined later.

$\mathscr{C}$ outputs $accept$ if all the conditions hold. Otherwise, $\mathscr{C}$ outputs $reject$.

**Definition 4.** $\mathscr{A}_1$ breaks a CLOSAS scheme with $(\varepsilon, q_c, q_r, q_s, q_p, q_h, q_{sig}, n, t)$ if and only if $\mathscr{C}$ outputs *accept* in the above game with a success probability greater than $\varepsilon$ within the execution time $t$, where $\mathscr{A}_1$ who does not know $msk$ can generate at most $q_c$ create-user queries, $q_r$ public-key-replace queries, $q_s$ secret-value-extract queries, $q_p$ partial-private-key-extract queries, $q_h$ random oracle queries and $q_{sig}$ signing queries, and $n$ is an upper bound for the number of signers included in the forgery output by $\mathscr{A}_1$.

**Definition 5.** A CLOSAS scheme is secure with $(\varepsilon, q_c, q_r, q_s, q_p, q_h, q_{sig}, n, t)$ if and only if there is no adversary $\mathscr{A}_1$ who breaks the CLOSAS scheme with $(\varepsilon, q_c, q_r, q_s, q_p, q_h, q_{sig}, n, t)$.

### 3.2.3 Game 2

This game is executed between $\mathscr{C}$ and $\mathscr{A}_2$.

**Setup** $\mathscr{C}$ runs the setup algorithm described in the previous section to obtain $param, msk$ and $mpk$. $\mathscr{C}$ gives $param, mpk$ and $msk$ to $\mathscr{A}_2$.

**Queries** $\mathscr{A}_2$ can access all the oracles described in section 3.2.1 and obtains the outputs.

[2]In the normal adversary, $y_i$ is required to be the original pubic key by $ID_i$. On the other hand, in the strong adversary, if $y_i$ is replaced, then the corresponding secret value $x_i$ is required as the additional input. In this paper, by the ability of the super adversary, the sign oracle require only $y_i$ even if $y_i$ is replaced.

**Forgery** $\mathscr{A}_2$ outputs a forgery $(\{ID_j^*\}_{j=1,\cdots,n}, \{m_j^*\}_{j=1,\cdots,n}, \psi_n^*, \sigma_n^*)$ and check that the following conditions hold.

- $\sigma_n^*$ is a valid signature on $\{m_j^*\}_{j=1,\cdots,n}$ in $\psi_n^*$ under $\{pk_j^*\}_{j=1,\cdots,n}$.
- Exactly one $ID_{i^*}^*$ who has never been queried for secret-value-extract oracle and public-key-replace oracle.
- Each $ID_i^*$ does not appear more than once in $\{ID_j^*\}_{j=1,\cdots,n}$.
- For $ID_{i^*}^*$, $m_{i^*}^* \notin \{m_{i^*}^{(1)}, \cdots, m_{i^*}^{(q_s)}\}$ or $\psi_{i^*}^* \notin \{\psi_{i^*}^{(1)}, \cdots, \psi_{i^*}^{(q_s)}\}$ holds.

$\mathscr{C}$ outputs *accept* if all the conditions hold. Otherwise, $\mathscr{C}$ outputs *reject*.

**Definition 6.** $\mathscr{A}_2$ breaks a CLOSAS scheme with $(\varepsilon, q_c, q_r, q_s, q_h, q_{sig}, n, t)$ if and only if $\mathscr{C}$ outputs *accept* in the above game with a success probability greater than $\varepsilon$ within the execution time $t$, where $\mathscr{A}_2$ can generate at most $q_c$ create-user queries, $q_r$ public-key-replace queries, $q_s$ secret-value-extract queries, $q_h$ random oracle queries and $q_{sig}$ signing queries, and $n$ is an upper bound for the number of signers.

**Definition 7.** A CLOSAS scheme is secure with $(\varepsilon, q_c, q_r, q_s, q_h, q_{sig}, n, t)$ if and only if there is no adversary $\mathscr{A}_2$ who breaks the CLOSAS scheme with $(\varepsilon, q_c, q_r, q_s, q_h, q_{sig}, n, t)$.

## 4 Proposed Scheme

In this section, we propose our CLOSAS scheme. In our scheme, we use state information $s$ similarly with the paper [11]. The state information is one-time information such as time-stamp, and is used to efficiently aggregate the data size of signatures according to [11]. In our scheme, *Signing* phase is run by each signer in turn, and the signature is implicitly aggregated in *Signing* phase instead of an aggregate phase in papers [6, 11, 35].

### 4.1 Construction

**Setup** A KGC generates a pairing parameter $(p, \mathbb{G}, \mathbb{G}_T, \mathrm{e})$. The KGC generates a generator $g \leftarrow \mathbb{G}$ and a random number $a \leftarrow \mathbb{Z}_p^*$. Then sets $A = g^a$, and chooses hash functions $H_1 : \{0,1\}^* \times \{0,1\} \rightarrow \mathbb{G}$, $H_2, H_3,: \{0,1\}^* \rightarrow \mathbb{G}$ and $H_4 : \{0,1\} \rightarrow \mathbb{Z}_p^*$. Finally, KGC outputs $(p, \mathbb{G}, \mathbb{G}_T,\ \mathrm{e}, g, H_1, H_2, H_3, H_4)$ as *param* and $A$ as *mpk*, and keeps $a$ to be secret as *msk*.

**Partial-Private-Key-Extract** Given signer's identity $ID_i$, KGC computes $g_{i,j} = H_1(ID_i, j)$ for $j = 0,1$ and then computes $g_{i,j}^a$. KGC sends $g_{i,j}^a$, $j = 0,1$, to $ID_i$ as his/her partial private key.

**User-Key-Gen** $ID_i$ generates a random number $t_i \leftarrow \mathbb{Z}_p^*$ and computes $T_i = g^{t_i}$. Then $ID_i$ outputs $T_i$ as his/her public key and keeps $t_i$ to be secret as his/her secret key.

**Set-Key** Given $g_{i,j}^a$ for $j = 0,1$ by KGC, $ID_i$ sets $(g_{i,0}^a, g_{i,1}^a, t_i)$ as his/her secret key $sk_i$ and $(ID_i, T_i)$ as its corresponding public key $pk_i$.

**Signing** Given $\{m_j\}_{j=1,\cdots,i-1}, \{ID_j\}_{j=1,\cdots,i-1}, \psi_{i-1}, \sigma_{i-1}, s$ by the previous signer, $ID_i$ first parses $\sigma_{i-1}$ as $(S_{i-1}, R_{i-1})$ and verifies that $\sigma_{i-1}$ is a valid signature on $\{m_j\}_{j=1,\cdots,i}$ in $\psi_{i-1}$ for $\{ID_j\}_{j=1,\cdots,i-1}$ by using verification algorithm with $n = i-1$ in this case. If not, $ID_i$ aborts the process. For the first signer (i.e. $ID_i = ID_1$), the above verification step is skipped and he/she sets $\psi_0 = \emptyset$, $S_0 = 1, R_0 = 1$ as the initial values. Then, $ID_1$ executes the following step similarly with the other signers.

For $ID_i$, $1 \le i \le n$, if the signature is valid, he/she sets $L_i = m_1 \parallel ID_1 \parallel \cdots \parallel m_i \parallel ID_i$. Then $ID_i$ computes $V = H_2(s)$, $W_i = H_3(s \parallel L_i)$, $c_i = H_4(s \parallel L_i)$, and generates a random number $r_i \leftarrow \mathbb{Z}_p^*$ and computes the following values:

$$S_i = V^{r_i} g_{i,0}^{a} \left(g_{i,1}^{a}\right)^{c_i} W_i^{t_i} \cdot S_{i-1}, \tag{1}$$

$$R_i = g^{r_i} \cdot R_{i-1}. \tag{2}$$

He/She sets $\sigma_i = (S_i, R_i)$ and sends $\{m_j\}_{j=1,\cdots,i}$, $\{ID_j\}_{j=1,\cdots,i}$, $\psi_i, \sigma_i, s$ to the next signer $ID_{i+1}$.

**Verification** Given $\{m_j\}_{j=1,\cdots,n}, \{ID_j\}_{j=1,\cdots,n}$, $\psi_n, \sigma_n, s$, A verifier parses $\sigma_n$ as $(S_n, R_n, s)$ and sets $L_j = m_1 \parallel ID_1 \parallel \cdots \parallel m_j \parallel ID_j$ for all $j$. Then he/she verifies that the following equation holds:

$$\mathrm{e}(S_n, g) \stackrel{?}{=} \mathrm{e}(V, R_n) \cdot \mathrm{e}\left(\prod_{j=1}^{n} g_{j,0} g_{j,1}^{c_j}, A\right) \cdot \prod_{j=1}^{n} \mathrm{e}\left(W_j, T_j\right), \tag{3}$$

where, for all $j$, $g_{j,l} = H_1(ID_j, l)$ for $l = 0,1$, $V = H_2(s)$, $W_j = H_3(s \parallel L_j)$, $c_j = H_4(s \parallel L_j)$.

### 4.2 Correctness

From the equations (1,2), the equation (3) can be written as follows:

$$\begin{aligned}
\mathrm{e}(S_n, g) &= \mathrm{e}\left(\prod_{j=1}^{n}\left(V^{r_j} g_{j,0}^{a} (g_{j,1}^{a})^{c_j} W_j^{t_j}\right), g\right) \\
&= \mathrm{e}\left(V, g^{\Sigma_{j=1}^{n} r_j}\right) \cdot \mathrm{e}\left(\prod_{j=1}^{n} g_{j,0} (g_{j,1})^{c_j}, g^{a}\right) \prod_{j=1}^{n} \mathrm{e}\left(W_j, g^{t_j}\right) \\
&= \mathrm{e}(V, R) \cdot \mathrm{e}\left(\prod_{j=1}^{n} g_{j,0} (g_{j,1})^{c_j}, A\right) \prod_{j=1}^{n} \mathrm{e}\left(W_j, T_j\right).
\end{aligned}$$

## 5 Discussion

### 5.1 Security Analysis

In this section, we discuss the security of the proposed scheme against adversaries described on section 3.2. In particular, when the adversary breaks the proposed scheme in each game, we construct an algorithm $\mathscr{B}$ to solve CDH problem by using the adversary.

**Theorem 8.** The proposed scheme is secure against type 1 of the adversary with $(\varepsilon, q_c, q_r, q_s, q_p, q_{h_1}, q_{h_2}, q_{h_3}, q_{h_4}, q_{sig}, n, t)$ if and only if $(t', \varepsilon')$-CDH assumption holds, where

$$\varepsilon' = \left(\varepsilon - \frac{q_{sig}(q_{sig}-1)}{2p}\right) \frac{27}{\left(q_p + q_{h_1} + q_{h_4} + (q_{h_1} + q_{h_2} + q_{h_4}) q_{sig}\right)^3} \cdot \frac{1}{e^3}, \tag{4}$$

$$t' = t + \mathscr{O}(q_{sig} + n(q_c + q_r + q_p + q_s + q_{h_1} + q_{h_2} + q_{h_3} + q_{h_4})) + \Psi, \tag{5}$$

where $\Psi$ is the computational time for the final result.

*Proof (Sketch).* The proof is given in appendix A. □

**Theorem 9.** The proposed scheme is secure against type 2 of the adversary with $(\varepsilon, q_c, q_s, q_{h_1}, q_{h_2}, q_{h_3}, q_{h_4}, q_{sig}, n, t)$ if and only if $(t', \varepsilon')$-CDH assumption holds, where

$$\varepsilon' = \left(\varepsilon - \frac{q_{sig}(q_{sig}-1)}{2p}\right)\frac{27}{(q_s+(q_{h_1}+q_{h_2}+q_{h_3})q_{sig})^3}\cdot\frac{1}{e^3}, \quad (6)$$

$$t' = t + \mathscr{O}(q_{sig}+n(q_c+q_r+q_s+q_{h_1}+q_{h_2}+q_{h_3}+q_{h_4})) + \Psi, \quad (7)$$

and $\Psi$ is the computational time for the final result.

*Proof (Sketch).* The proof is given in appendix B. □

## 5.2 Construction Resisting the DoD Attack

Liu et al.[19] have pointed out a problem in distributing public keys in a certificateless setting. Suppose an adversary replace a public key of any user with other faked public key. Then an encryptor who cannot detect the replacement, certificateless property, performs the encryption under the faked public key. Such data encrypted under the faked public key cannot be decrypted by the user correctly because the user does not know a secret value corresponding to the replaced faked public key. This attack is called Denial of Decryption (DoD) attack. In order to prevent this attack, they have proposed a method to guarantee the validity of a public key without the interaction with any trusted authority, i.e. *self-generated-certificate*. In this method, each user guarantees the validity of a public key by generating a certificate, signature, under a secret key corresponding to the public key.

DoD attack may also occur in digital signature scheme in that a digital signature generated by any user is maliciously rejected by the replacement of its own public key. In this approach, Wu proposed a digital signature scheme with self-generated-certificate[50]. Since the user can detect the replacement of the public key by the verification with the self-generated-certificate, it can resist against malicious rejection of signature. However, the construction with the self-generated-certificate cannot achieve level-3 security. In particular, the malicious KGC can still impersonate any user by generating a pair of a secret key and a public key and its corresponding self-generated-certificate by him-/herself.

The notion of self-generated-certificate can be applied to our scheme. In paper [33], which is a previous version of this work, we proposed a CLOSAS scheme with self-generated-certificate. In this section, we give the detail of the construction. Although the following construction cannot be achieved level-3 security, the proposed scheme becomes more secure in the sense that the scheme resist DoD attack.

### 5.2.1 Construction

**Setup** This algorithm is same as the proposed scheme in section 4.1.

**Partial-Private-Key-Extract** Given signer's identity $ID_i$, KGC computes $g_{i,j} = H_1(ID_i, j)$ for $j = 0, 1$ and then computes $g^a_{i,j}$. KGC sends $g^a_{i,j}$ for $j = 0, 1$ to $ID_i$ as his/her partial private key.

**User-Key-Gen** A signer $ID_i$ generates random numbers $t_{i,0}, t_{i,1} \leftarrow \mathbb{Z}^*_p$ and computes $T_{i,0} = g^{t_{i,0}}, T_{i,1} = g^{t_{i,1}}$. Then $ID_i$ outputs $T_{i,0}, T_{i,1}$ as his/her public key and keeps $t_{i,0}, t_{i,1}$ to be secret as his/her secret key.

**Set-Key** Given $g_{i,j}^a$ and $t_{i,j}$ for $j = 0, 1$ by KGC, $ID_i$ sets $(g_{i,0}^a, g_{i,1}^a, t_{i,0}, t_{i,1})$ as his/her secret key $sk_i$, and generates a random number $r_i'$ and state information $s_i$. Then $ID_i$ sets $m_i' := ID_i \parallel T_{i,1}$ and computes as follows:

$$\begin{aligned} S_i' &= V_i^{r_i'} g_{i,0}^a \left(g_{i,1}^a\right)^{c_i'} W_i'^{t_{i,0}}, & (8) \\ R_i' &= g^{r_i'}, & (9) \end{aligned}$$

where $V_i = H_2(s_i)$, $W_i = H_3(s_i \parallel m_i')$ and $c_i' = H_4(s_i \parallel m_i')$. $ID_i$ sets $\sigma_i' = (S_i', R_i', s_i)$ and $(ID_i, T_{i,0}, T_{i,1}, \sigma_i')$ as its corresponding public key $pk_i$.

**Signing** Given $\{m_j\}_{j=1,\cdots,i-1}, \{ID_j\}_{j=1,\cdots,i-1}, \psi_{i-1}$, $\sigma_{i-1}$ by the previous signer, $ID_i$ first parses $\sigma_{i-1}$ as $(S_{i-1}, R_{i-1}, s)$ and verifies that $\sigma_{i-1}$ is a valid signature on $\{m_j\}_{j=1,\cdots,i}$ in $\psi_{i-1}$ for $\{ID_j\}_{j=1,\cdots,i}$ by using verification algorithm with $n = i-1$ in this case. If not, $ID_i$ aborts the process. For the first signer (i.e. $ID_i = ID_1$), the above verification step is skipped and he/she sets $\psi_0 = \emptyset, S_0 = 1, R_0 = 1$ as the initial values. Then, $ID_1$ executes the following step similarly with the other signers.

For $ID_i$, $1 \le i \le n$, if the signature is valid, he/she sets $L_i = m_1 \parallel ID_1 \parallel \cdots \parallel m_i \parallel ID_i$. Then $ID_i$ computes $V = H_2(s)$, $W_i = H_3(s \parallel L_i)$, $c_i = H_4(s \parallel L_i)$, and generates a random number $r_i \leftarrow \mathbb{Z}_p^*$ and computes the following values:

$$\begin{aligned} S_i &= V^{r_i} g_{i,0}^a \left(g_{i,1}^a\right)^{c_i} W_i^{t_{i,1}} \cdot S_{i-1}, & (10) \\ R_i &= g^{r_i} \cdot R_{i-1}. & (11) \end{aligned}$$

He/She sets $\sigma_i = (S_i, R_i, s)$ and $\psi_i = \psi_{i-1} \parallel ID_i$, and sends $\{m_j\}_{j=1,\cdots,i}$, $\{ID_j\}_{j=1,\cdots,i}, \psi_i, \sigma_i$ to the next signer $ID_{i+1}$.

**Verification** Given $\{m_j\}_{j=1,\cdots,n}, \{ID_j\}_{j=1,\cdots,n}$, $\psi_n, \sigma_n$, A verifier verifies that, for $\{ID_j\}_{j=1,\cdots,n}$, the public key $pk_j$ is correct. In particular, the verifier parses the signers' self-generated-certificates $\sigma_j'$ in $pk_j$ as $(S_j', R_j', s_j)$ for $j = 1, \cdots, n$, and set $m_j' := ID_j \parallel T_{j,1}$. Then, he/she computes as follows:

$$\mathrm{e}(S_j', g) \stackrel{?}{=} \mathrm{e}(V_j, R_j') \cdot \mathrm{e}\left(g_{j,0} g_{j,1}^{c_j'}, A\right) \cdot \mathrm{e}\left(W_j', T_{j,0}\right), \tag{12}$$

where, for all $j$, $g_{j,l} = H_1(ID_j, l)$ for $l = 0, 1$, $V_j = H_2(s_j)$, $W_j' = H_3(s_j \parallel m_j')$ and $c_j' = H_4(s_j \parallel m_j')$. If the above equation holds for all signers, then the verifier parses $\sigma_n$ as $(S_n, R_n, s)$ and sets $L_j = m_1 \parallel ID_1 \parallel \cdots \parallel m_j \parallel ID_j$ for all $j$. Then he/she verifies that the following equation holds:

$$\mathrm{e}(S_n, g) \stackrel{?}{=} \mathrm{e}(V, R_n) \cdot \mathrm{e}\left(\prod_{j=1}^{n} g_{j,0} g_{j,1}^{c_j}, A\right) \cdot \prod_{j=1}^{n} \mathrm{e}\left(W_j, T_{j,1}\right), \tag{13}$$

where, for all $j$, $V = H_2(s)$, $W_j = H_3(s \parallel L_j)$, and $c_j = H_4(s \parallel L_j)$. If the above equation holds, the verifier outputs *accept*. Otherwise, he/she outputs *reject*.

**Theorem 10.** A signature in the proposed scheme described in section 4 is existentially unforgeable if and only if a self-generated-certificate in the scheme in [33] is existentially unforgeable.

*Proof (Sketch).* Intuitively, if an adversary who can forge a self-generated-certificate exists, then the adversary can also forge an aggregate signature in the proposed scheme in this paper by using the forged self-generated-certificate as a signature of the target signer. This result conflicts with the theorems described in the previous section. □

Table 1: Evaluation of the schemes

| | Signing Cost for $i$-th Signer | Verification Cost | Signature Size | Type of Scheme | Certificateless Property |
|---|---|---|---|---|---|
| Boldyreva et al.[5] | $\mathscr{H}+3E(1)$ $+E(2(i-1))+3$ | $3\mathscr{P}+\mathscr{H}$ $+E(n)+E(2n)$ | $2l(p)$ | Ordered | No |
| Fischlin et al.[10] | $\mathscr{P}+2\mathscr{H}+E(1)+1$ | $n\mathscr{H}+n^2\mathscr{P}$ | $3l(p)$ | Ordered | No |
| Wang et al.[29] | $2\mathscr{H}+\mathscr{R}+E(2)+1$ | $(n+1)\mathscr{P}+nE(2)$ $+3n\mathscr{H}+2(n-1)$ | $(n+1)l(p)$ | Ordered | No |
| Yamamoto et al.[31] | $\mathscr{H}+E(1)+i$ | $(n+1)\mathscr{P}+n\mathscr{H}$ | $l(p)$ | Ordered | No |
| Zhang et al.[35] | $4\mathscr{H}+E(1)+E(5)$ | $5\mathscr{P}+4E(n)$ $+(4n+3)\mathscr{H}$ | $2l(p)$ | General | Yes |
| Our Scheme | $3\mathscr{H}+E(1)$ $+E(4)+2$ | $(3+n)\mathscr{P}+E(n)$ $+(4n+1)\mathscr{H}$ | $2l(p)$ | Ordered | Yes |

## 6 Evaluation

We compare the performance of the proposed scheme with some existing schemes with respect to the signing cost, the verification cost, the signature size, type of the scheme and certificateless property. The result is shown in table 1. For the evaluation of the signing cost and the verification cost, we adopt the same method with [28]. We denote by $\mathscr{P}$ the computational cost of pairing, by $\mathscr{H}$ the computational cost of hash functions, by $\mathscr{R}$ the ratio of the computational cost of multiplication in $\mathbb{Z}_p^*$ to that of multiplication in $\mathbb{F}_p$ and by $E(n) := (\frac{n}{2}+1)l(p)-1$ the required number of modulo-$p$ multiplication for computing $g_1^{a_1}\cdots g_n^{a_n}$ with $g_i \in \mathbb{Z}_p^*$ and $a_i \in \mathbb{Z}_p$, where $l(p)$ denotes the binary length of $p$. For the type of the scheme, Ordered means ordered sequential type and General means general type as described in section 1.2.2. Finally, for certificateless property, Yes means a certificateless scheme.

As shown in Table 1, our scheme has the same signature size as those in [5, 35], and the verification cost is similar to that in [29]. In comparison to the scheme in [35], our proposed scheme is efficient in the signing cost and hence the scheme is suitable for devices of low computational power such as mobile phone. In addition, to the best of our knowledge, our proposed scheme is the only ordered sequential aggregate signature scheme in the certificateless setting.

## 7 Conclusion

Certificateless cryptosystem is a cryptosystem that overcomes abuses of key escrow of KGC, and we proposed a certificateless ordered sequential aggregate signature scheme. To the best of our knowledge, our proposed scheme is the first OSAS scheme in a certificateless setting. Although the computational cost for pairing computation in our scheme is linear with respect to the number of signers, our scheme achieved the full aggregation with the security proof against the strongest adversary, super adversary, in the random oracle model. On the subject of the security proof, the super adversaries are adversaries who can access a black box knowledge extractor which extracts a secret key from a corresponding public key without detecting by a target signer, and we have also given a security model that captures both the super adversaries and the security requirements in CLOSAS scheme. Namely, our defined model takes into account the cryptographically strongest insider threats about the security in CLOSAS scheme, and through the security analysis based on this model, we proved that our scheme can resist a forgery of signatures as long as at least one value in full secret key is kept secret.

In future work, we plan to extend our scheme so as to achieve a fixed number of pairing computation with respect to the number of signers and to prove the security against an actively malicious KGC described in section 1.2.1, i.e. achieving Girault's level-3 security[41].

## Acknowledgments

This research has been supported by Interchange Association Japan and Support Center for Advanced Telecommunications Technology Research. We would like to thank for their great support. We would like to thank anonymous reviewers for their invaluable comments also.

## References

[1] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, February 1978.

[2] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proc. of International Cryptograpy Conference (CRYPTO'84), Santa Barbara, USA*, vol. 196. Springer-Verlag, August 1987, pp. 47–53.

[3] A. K. Lenstra, J. P. Hughes, M. Augier, J. W. Bos, T. Kleinjung, and C. Wachter, "Ron was rong, whit is right," Cryptology ePrint Archive: Listing for 2012, pp. 1–17, February 2012, http://eprint.iacr.org/2012/064.

[4] S. S. Al-Riyami and K. G. Paterson, "Certificateless public key cryptography," in *Proc. of the 9th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT'03), Taipei, Taiwan*, vol. 2894. Spirnger-Verlag, November-December 2003, pp. 452–473.

[5] A. Boldyreva, C. Gentry, A. O'Neill, and D. H. Yum, "Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing," in *Proc. of the 14th ACM Conference on Computer and Communication Security (CCS'07), Alexandria, USA*. ACM, October-November 2007, pp. 276–285.

[6] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and verifiably encrypted signatures from bilinear maps," in *Proc. of the 22th Theory and Applications of Cryptographic Techniques (EUROCRYPT'03), Warsaw, Poland*, vol. 2656. Springer-Verlag, May 2003, pp. 416–432.

[7] K. Brogle, S. Goldberg, and L. Reyzin, "Sequential aggregate signatures with lazy verification," Cryptology ePrint Archive: Listing for 2011, pp. 1–30, May 2011, http://eprint.iacr.org/2011/222.

[8] M. Burmester, Y. Desmedt, H. Doi, M. Mambo, E. Okamoto, M. Tada, and Y. Yoshifuji, "A structured elgamal-type multisignature scheme," in *Proc. of the 9th International Conference on Theory and Practice of Public-Key Cryptography (PKC'00), Melbourne, Australia*, vol. 1751. Springer-Verlag, January 2000, pp. 466–483.

[9] B. Dou, H. Zhang, C. Xu, and M. Han, "Identity-based sequential aggregate signature from rsa," in *Proc. of the 4th ChinaGrid Annual Conference (ChinaGrid'09), Yantai, China*. IEEE, August 2009, pp. 123–127.

[10] M. Fischlin, A. Lehmann, and D. Schrōder, "History-free sequential aggregate signatures," Cryptology ePrint Archive: Listing for 2011, pp. 1–18, May 2012, http://eprint.iacr.org/2011/231.

[11] C. Gentry and Z. Ramzan, "Identity-based aggregate signatures," in *Proc. of the 9th International Conference on Theory and Practice of Public-Key Cryptography (PKC'06) New York, USA*, vol. 3958. Springer-Verlag, April 2006, pp. 257–273.

[12] Z. Gong, Y. Long, X. Hong, and K. Ghen, "Practical certificateless aggregate signatures from bilinear maps," *Journal of Information Science and Engineering*, vol. 26, no. 6, pp. 2093–2106, 2010.

[13] C. Han, H. Zhang, B. Zhang, and Y. Yang, "A structured multi-signature scheme and its security proof," in *Proc. of IEEE International Conference on Information Theory and Information Security (ICITIS'10), Beijing, China*. IEEE, December 2010, pp. 345–348.

[14] K. Itakura and K. Nakamura, "A public-key cryptosystem suitable for digital multi-signatures," *NEC Research and Development*, vol. 71, pp. 1–8, 1983.

[15] K. Kawauchi and M. Tada, "On the security and the efficiency of multi-signature schemes based on a trapdoor one-way permutation," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E88-A, no. 5, pp. 1274–1282, May 2005.

[16] X. Li, L. Zhang, and S. Li, "Proxy structured multisignature scheme from bilinear pairing," in *Proc. of the 2nd International Symposium on Parallel and Distributed Processing and Applications (ISPA'04), Hong Kong, China*, vol. 3358. Springer-Verlag, December 2004, pp. 705–714.

[17] C.-Y. Lin, T.-C. Wu, and F. Zhang, "A structured multisignature scheme from the gap diffie-hellman group," Cryptology ePrint Archive: Listing for 2003, pp. 1–5, May 2003, http://eprint.iacr.org/2003/090.

[18] J. K. Liu, J. Baek, and J. Zhou, "Certificate-based sequential aggregate signature," in *Proc. of the 2nd ACM Conference on Wireless Network Security (WiSec'09), Zurich, Switzerland*. ACM, March 2009, pp. 21–28.

[19] J. K. Liu, M. H. Au, and W. Susilo, "Self-generated certificate public key cryptography and certificateless signature / encryption scheme," in *Proc. of the 2nd ACM symposium on Informatoin, Computer and Communications Security (ASIACCS'07), Singapole*. ACM, March 2007, pp. 273–283.

[20] A. Lysyanskaya, S. Micali, L. Reyzin, and H. Shacham, "Sequential aggregate signatures from trapdoor permutations," in *Proc. of the 23th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'04), Interlaken, Switzerland*, vol. 3027. Springer-Verlag, May 2004, pp. 74–90.

[21] S. Lu, R. Ostrovsky, A. Sahai, H. Shacham, and B. Waters, "Sequential aggregate signatures and multisignatures without random oracle," in *Proc. of the 25th Theory and Applications of Cryptographic Techniques (EUROCRYPT'06), Petersburg, Russia*, vol. 4004. Springer-Verlag, May 2006, pp. 465–485.

[22] S. Mitomi and A. Miyaji, "A general model of multisignature schemes with message flexibility,," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E84-A, no. 10, pp. 2488–2499, October 2001.

[23] Y. Muxiang, S. Li, L. Jun, and H. Fan, "Secure order-specified multisignature scheme based on dsa," *Wuhan University Journal of Natural Sciences*, vol. 11, no. 6, pp. 1613–1616, November 2006.

[24] G. Neven, "Efficient sequential aggregate signed data," *IEEE Transactions on Information Theory*, vol. 57, no. 3, pp. 1803–1815, March 2011.

[25] K. Ohta and T. Okamoto, "Multi-signature schemes secure against active insider attacks," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E82-A, no. 1, pp. 21–31, January 1999.

[26] S. S. D. Selvi, S. S. Vivek, and C. P. Rangan, "A suite of identity based aggregate signatures and a multi-signature scheme from rsa," Cryptology ePrint Archive: Listing for 2010, pp. 1–12, September 2010, http://eprint.iacr.org/2010/493.

[27] ——, "Efficient and provably secure identity based aggregate signature schemes," Cryptology ePrint Archive: Listing for 2010, pp. 1–12, 2010, http://eprint.iacr.org/2010/461.

[28] M. Tada, "A secure multisignature scheme with signing order verifiability," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E86-A, no. 1, pp. 73–88, January 2003.

[29] L. Wang, E. Okamoto, Y. Miao, T. Okamoto, and H. Doi, "An id-sp-m4m scheme and its security analysis," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E90-A, no. 1, pp. 91–100, January 2007.

[30] W. Wu, Y. Mu, W. Susilo, and X. Huang, "Certificate-based signatures revisited," *Journal of Universal Computer Science*, vol. 15, no. 8, pp. 1659–1684, August 2009.

[31] D. Yamamoto and W. Ogata, "A general model of structured multisignatures with message flexibility," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E90-A, no. 1, pp. 83–90, January 2007.

[32] N. Yanai, E. Chida, and M. Mambo, "A secure structured multisignature scheme based on a non-commutative ring homomorphism," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E94-A, no. 6, pp. 1346–1355, June 2011.

[33] N. Yanai, R. Tso, M. Mambo, and E. Okamoto, "Certificateless ordered sequential aggregate signature scheme," in *Proc. of the 3rd International Conference on Intelligent Networking and Collaborative Systems*

*(INCoS'11), Fukuoka, Japan*. IEEE, November-December 2011, pp. 662–667.

[34] J. Zhang, "An improved structured multi-signature scheme," in *Proc. of the 2nd International Conference on Information Management and Engineering (ICIME'10), Chengdu, China*. IEEE, April 2010, pp. 54–58.

[35] L. Zhang, B. Qin, Q. Wu, and F. Zhang, "Efficient many-to-one authentication with certificateless aggregate signatures," *Computer Networks*, vol. 54, no. 14, pp. 2482–2491, October 2010.

[36] M. Zhao, S. Smith, and D. Nicol, "Aggregated path authentication for efficient bgp security," in *Proc. of the 12th ACM Conference on Computer and Communications Security (CCS'05), Alexandria, USA*. ACM, November 2005, pp. 128–138.

[37] H. Zhu, F. Bao, and R. H. Deng, "Sequential aggregate signatures working over independent homomorphic trapdoor," in *Proc. of the 7th International Conference (ICICS'05), Beijing, China*, vol. 3783. Springer-Verlag, December 2005, pp. 207–219.

[38] S. Kent, C. Lynn, and K. Seo, "Secure border gateway protocol," *IEEE Journal of Selected Areas in Communications*, vol. 18, no. 4, pp. 582–592, 2000.

[39] X. Huang, Y. Mu, W. Susilo, D. Wong, and W. Wu, "Certificateless signature revisited," in *Proc. of the 12th Australasian Conference on Information Security and Privacy (ACISP'07), Townsville, Australia,*, vol. 4586. Springer-Verlag, July 2007, pp. 308–322.

[40] R. Castro and R. Dahab, "Efficient certificateless signatures suitable for aggregation," Cryptology ePrint Archive: Listing for 2007, pp. 1–24, December 2007, http://eprint.iacr.org/2007/454.

[41] M. Girault, "Self-certified public keys," in *Proc. of the 10th Theory and Applications of Cryptographic Techniques (EUROCRYPT'91), Brighton, UK*, vol. 547. Spriger-Verlag, April 1991, pp. 490–497.

[42] B. G. Kang, J. H. Park, and S. G. Hahn, "A certificate-based signature scheme," in *Proc. of the Cryptographers' Track at the RSA Conference (CT-RSA'04), San Francisco, USA*, vol. 2964. Springer-Verlag, February 2004, pp. 99–111.

[43] J. Li, X. Huang, Y. Mu, W. Susilo, and Q. Wu, "Certificate-based signature: Security model and efficient construction," in *Proc. of the 4th European Workshop on Public Key Infrastructure (EuroPKI'07), Palma de Mallorca*, vol. 4582. Springer-Verlag, June 2007, pp. 110–125.

[44] M. Bellare and P. Rogaway, "Random oracle are practical: A paradigm for designing efficient protocols," in *Proc. of the 1st ACM Conference on Computer and Communications Security (CCS'93), Fairfax, USA*. ACM, November 1993, pp. 62–73.

[45] R. Canetti, O. Goldreich, and S. Halevi, "The random oracle methodology, revisited," *Journal of the ACM*, vol. 51, no. 4, pp. 557–594, July 2004.

[46] Y. Rekhter and T. Li, "A border gateway protocol 4 (bgp-4)," RFC 1771, March 1995, http://www.ietf.org/rfc/rfc1771.txt.

[47] S. Kent, "Securing the border gateway protocol: A status update," in *Proc. of the 7th IFIP TC-6 TC-11 Conference on Communications and Multimedia Security (CMS'03), Torino, Italy*, vol. 2828. Springer-Verlag, October 2003, pp. 40–53.

[48] J. Y. Hwang, D. H. Lee, and M. Yung, "Universal forgery of the identity-based sequential aggregate signature scheme," in *Proc. of the 4th ACM Conference on Computer and Communications Security (ASIACCS'09), Sydney, Australia*. ACM, March 2009, pp. 157–160.

[49] H. Xiong, Q. Wu, and Z. Chen, "Strong security enabled certificateless aggregate signatures applicable to mobile computation," in *Proc. of the 3rd International Conference on Intelligent Networking and Collaborative Systems (INCoS'11), Fukuoka, Japan*. IEEE, November-December 2011, pp. 92–99.

[50] C. Wu, "Self-generated-certificate digital signature," in *Proc. of the 4th International Conference on Generic and Evolutionary Computing (ICGEC'10), Shenzhen, China*. IEEE, December 2010, pp. 379–382.

**Naoto Yanai** received B.Eng. degree in electrical engineering from Ichinoseki National College of Technology, Japan, in 2009 and M.S. Eng. in graduate school of systems and information engineering from Univerisity of Tsukuba, Japan, in 2011. He has recently joined Dr. course in systems and information engineering in University of Tsukuba, Japan.

**Rayling Tso** is an assistant professor of Computer Science at National Chengchi University, Taiwan. He received his PhD degree in Systems and Information Engineering from University of Tsukuba, Japan in 2006. His research interests are mainly in cryptography including secret sharing, key agreement, digital signatures, certificateless cryptosystems, etc. Recently, his research activities are focused on certificateless signatures and digital signatures that providing privacy protection.

**Masahiro Mambo** received a B.Eng. degree from Kanazawa University, Japan, in 1988 and M.S.Eng. and Dr.Eng. degrees in electronic engineering from Tokyo Institute of Technology, Japan in 1990 and 1993, respectively. After working at Japan Advanced Institute of Science and Technology, JAIST, Tohoku University and University of Tsukuba, he joined Kanazawa University in 2011. He is currently a professor of Faculty of Electrical and Computer Engineering, Institute of Science and Engineering. His research interests include information security, software protection and privacy protection.

**Eiji Okamoto** received his B.S., M.S. and Ph.D degrees in electronics engineering from the Tokyo Institute of Technology in 1973, 1975 and 1978, respectively. He worked and studied communication theory and cryptography for NEC central research laboratories since 1978. In 1991 he became a professor at Japan Advanced Institute of Science and Technology, then at Toho University. Now he is a professor at Faculty of Engineering, Information and Systems, University of Tsukuba. His research interests are cryptography and information security. He is a member of IEEE and a coeditor-in-chief of Internatinal Journal of Information Security.

## A Proof of Theorem 8

This proof is based on the security proof in paper [11], and we define a probability $\delta$ to set 1 for tossing a coin. To complete the proof, we finally determine a concrete value of $\delta$.

Given a CDH challenge $(g, g^a, g')$, $\mathscr{B}$ who tries to solve CDH problem generates a pairing parameter $(p, \mathbb{G}, \mathbb{G}_T, \mathrm{e})$, and sets $mpk = g^a$ and a certification list $\mathscr{L} = \emptyset$. This means that $\mathscr{B}$ implicitly sets $a$ as $msk$. Then $\mathscr{B}$ sets $ID$-list $[\cdot,\cdot]$ $H_1$-list $[\cdot,\cdot,\cdot,\cdot,\cdot,\cdot]$, $H_2$-list $[\cdot,\cdot,\cdot]$, $H_3$-list $[\cdot,\cdot,\cdot,\cdot,\cdot]$ and $H_4$-list $[\cdot,\cdot,\cdot,\cdot,\cdot,\cdot]$ as empty, and run $\mathscr{A}$ with $g, g^a$ as input. Here, without loss of generality, we assume that $\mathscr{B}$ executes $H_1$-query and $H_2$-query before executing $H_3$-query and $H_4$-query, $H_1$-query before executing the create-user query and each random oracle query before executing the signing oracle query.

$H_1$**-query** Given $ID_i$ generated by $\mathscr{A}$, check that $H_1$-list includes $ID_i$. If so, return $H_1(ID_i, j)$ from $H_1$-list, where $j = 0, 1$. Otherwise, toss a coin $H_1\text{-}coin_i \leftarrow \{0,1\}$ with probability $\delta$. If $H_1\text{-}coin_i = 0$, generate $\alpha_{i,0}, \alpha_{i,1} \leftarrow \mathbb{Z}_p$ and set $\alpha'_{i,0} = \alpha'_{i,1} = 0$. Otherwise, generate $\alpha_{i,0}, \alpha_{i,1}, \alpha'_{i,0}, \alpha'_{i,1} \leftarrow \mathbb{Z}_p$. Set $H_1(ID_i, j) = (g^{\alpha_{i,j}} g'^{\alpha'_{i,j}})$, and register $(ID_i, H_1\text{-}coin_i, \alpha_{i,0}, \alpha_{i,1}, \alpha'_{i,0}, \alpha'_{i,1})$ on $H_1$-list. Return $H_1(ID_i, j)$, $j = 0, 1$.

$H_2$**-query** Given $s$ generated by $\mathscr{A}$, check that $H_2$-list includes $s$. If so, return $H_2(s)$ from $H_2$-list. Otherwise, toss a coin $H_2\text{-}coin_k \leftarrow \{0,1\}$ and generate $\beta \leftarrow \mathbb{Z}_p^*$. If $H_2\text{-}coin_k = 0$, set $V = g'^{\beta}$ as $H_2(s)$. Otherwise, set $V = g^{\beta}$ as $(H_2(s)$. Register $(s, H_2\text{-}coin_k, \beta)$ on $H_2$-list and return $H_2(s)$.

$H_3$**-query** Given $s \parallel L_i$ generated by $\mathscr{A}$, check that $H_3$-list includes $s \parallel L_i$. If so, return $H_3(s \parallel L_i)$ from $H_3$-list. Otherwise, generate $\gamma \leftarrow \mathbb{Z}_p^*$ and set $H_3(s \parallel L_i) = g^{\gamma}$. Register $(s, ID_i, m_i, L_i, \gamma)$ on $H_3$-list and return $H_3(s \parallel \psi_i)$.

$H_4$**-query** Given $s \parallel L_i$ generated by $\mathscr{A}$, check that $H_4$-list includes $s \parallel L_i$. If so, return $H_4(s \parallel L_i)$ from $H_4$-list. Otherwise, toss a coin $H_4\text{-}coin_l \leftarrow \{0,1\}$. If $H_4\text{-}coin_l = 0$, check that $H_1\text{-}coin_i = H_2\text{-}coin_k = 1$ for $s \parallel L_i$. If so, check that $s \parallel L_i \neq s \parallel L'_i$ exists with $ID_i = ID'_i$. If so, aborts. Otherwise, set $H_4(s \parallel \psi_i) = -\frac{\alpha'_{i,0}}{\alpha'_{i,1}}$. If $H_1\text{-}coin_i = H_2\text{-}coin_k = H_4\text{-}coin_l = 1$, check that $(s, m_i, ID_i) = (s', m'_i, ID'_i)$ and $\psi_i \neq \psi'_i$ exists. If so, aborts. Otherwise, set $d_{(i,k,l)} = 0$. If none of the above, generate $d_{(i,k,l)} \leftarrow \mathbb{Z}_p^*$. Set $H_4(s \parallel \psi_i) = d_{(i,k,l)}$. Register $(s, ID_i, m_i, L_i, H_4\text{-}coin_l, d_{(i,k,l)})$ on $H_4$-list and return $H_4(s \parallel \psi_i)$.

**Create-User** Given $ID_i$ generated by $\mathscr{A}$, check that $\mathscr{L}$ includes $ID_i$. If so, return $(ID_i, T_i)$ from $\mathscr{L}$. Otherwise, retrieve $H_1(ID_i, j)$ for $j = 0, 1$ from $H_1$-list as $g_{i,j}$, and generate $t_i \leftarrow \mathbb{Z}_p$. Set $T_i = g^{t_i}$, and register $(ID_i, T_i)$ in $\mathscr{L}$, $(ID_i, t_i)$ in $ID$-list. Return $H_1(ID_i, j)$ for $j = 0, 1$ and $T_i$.

**Partial-Private-Key-Extract** Given $ID_i$ generated by $\mathscr{A}$, check that $\mathscr{L}$ includes $ID_i$. If not, nothing will be output. Otherwise, check that $H_1\text{-}coin_i = 1$ holds. If so, abort. Otherwise, set $g^a_{i,j} = (g^a)^{\alpha_{i,j}}$ and return $g^a_{i,j}$ where $j = 0, 1$.

**Public-Key-Replace** Given $ID_i$ and $T'_i$ generated by $\mathscr{A}$, re-register $(ID_i, T'_i)$ in $\mathscr{L}$ and $(ID_i, nil)$, where $nil$ means an unknown value for $\mathscr{B}$.

**Secret-Value-Extract** Given $ID_i$ generated by $\mathscr{A}$, check that $\mathscr{L}$ includes $ID_i$. If not, nothing will be output. Otherwise, return $t_i$ from $\mathscr{L}$. Here, if the secret value corresponding to $ID_i$ in $ID$-list is $nil$, then nothing will be output.

**Signing** Given $\{m_j\}_{j=1,\cdots,i}, \{ID_j\}_{j=1,\cdots,i}, \psi_i, \sigma_i, s$ generated by $\mathscr{A}$, check that $H_1$-$coin_i$, $H_2$-$coin_k$ and $H_4$-$coin_l$. If $H_1$-$coin_i = H_2$-$coin_k = H_4$-$coin_l = 1$, abort. Otherwise, compute a signature as follows. In the case that $H_1$-$coin_i = 0$, generate a random number $r \leftarrow \mathbb{Z}_p^*$ and pick the latest public key $T_i$ of $ID_I$ from $\mathscr{L}$, which may be the original public key generated from Create-User or a false public key replaced by the adversary. Then, compute as follows:

$$\begin{aligned} S_i &= V^r (g^a)^{\alpha_{i,0}} (g^a)^{\alpha_{i,1} c_i} (T_i)^{\gamma} \cdot S_{i-1}, && (14)\\ R_i &= g^r \cdot R_{i-1}, && (15) \end{aligned}$$

where $V, \gamma$ and $c_i$ are retrieved from $H_2$-$list$,$H_3$-$list$ and $H_4$-$list$. These values become a valid signature on $\{m_j\}_{j=1,\cdots,i}$ in $\psi_i$ for $\{ID_j\}_{j=1,\cdots,i}$. In the case that $H_1$-$coin_i = 1 \wedge H_2$-$coin_i = 0$, compute as follows:

$$\begin{aligned} S_i &= \left(g'^{\beta}\right)^r (g^a)^{\alpha_{i,0}} (g^a)^{\alpha_{i,1} c_i} (T_i)^{\gamma} \cdot S_{i-1}, && (16)\\ R_i &= g^r (g^a)^{-\frac{\alpha'_{i,0}+\alpha'_{i,1}c_i}{\beta}} \cdot R_{i-1}, && (17) \end{aligned}$$

where $\beta, \gamma$ and $c_i$ are retrieved from $H_2$-$list$,$H_3$-$list$ and $H_4$-$list$. These values become a valid signature since they can be written as follows:

$$\begin{aligned} S_i &= \left(g'^{\beta}\right)^r (g^a)^{\alpha_{i,0}} (g^a)^{\alpha_{i,1} c_i} (T_i)^{\gamma} \cdot S_{i-1} \frac{(g'^a)^{\alpha'_{i,0}+\alpha'_{i,1}c_i}}{(g'^a)^{\alpha'_{i,0}+\alpha'_{i,1}c_i}} \\ &= \left(g'^{\beta}\right)^{r-a\frac{\alpha'_{i,0}+\alpha'_{i,1}c_i}{\beta}} \left(g^{\alpha_{i,0}} g'^{\alpha'_{i,0}}\right)^a \left(g^{\alpha_{i,1}} g'^{\alpha'_{i,1}}\right)^{ac_i} W_i^{t_i} S_{i-1}, && (18)\\ R_i &= g^{r-a\frac{\alpha'_{i,0}+\alpha'_{i,1}c_i}{\beta}} \cdot R_{i-1}. && (19) \end{aligned}$$

In the case that $H_1$-$coin_i = H_2$-$coin_i = 1 \wedge H_4$-$coin_i = 0$, compute as follows:

$$\begin{aligned} S_i &= \left(g^{\beta}\right)^r (g^a)^{\alpha_{i,0}} (g^a)^{\alpha_{i,1}(-\frac{\alpha'_{i,0}}{\alpha'_{i,1}})} (T_i)^{\gamma} \cdot S_{i-1}, && (20)\\ R_i &= g^r \cdot R_{i-1}. && (21) \end{aligned}$$

These values become a valid signature since they can be written as follows:

$$\begin{aligned} S_i &= \left(g^{\beta}\right)^r (g^a)^{\alpha_{i,0}} (g^a)^{\alpha_{i,1}(-\frac{\alpha'_{i,0}}{\alpha'_{i,1}})} (T_i)^{\gamma} \cdot S_{i-1} \left(g'^{\alpha'_{i,0}}\right)^a \left(g^{\prime\alpha'_{i,1}(-\frac{\alpha'_{i,0}}{\alpha'_{i,1}})}\right)^a \\ &= V^r \left(g^{\alpha_{i,0}} g'^{\alpha'_{i,0}}\right)^a \left(g^{\alpha_{i,1}} g'^{\alpha'_{i,1}}\right)^{a(-\frac{\alpha'_{i,0}}{\alpha'_{i,1}})} W_i^{t_i} \cdot S_{i-1}, && (22)\\ R_i &= g^r \cdot R_{i-1}. && (23) \end{aligned}$$

**Output** Given a forgery $(\{ID_j^*\}_{j=1,\cdots,n}, \{m_j^*\}_{j=1,\cdots,n}, \psi_n^*, \sigma_n^*)$ output by $\mathscr{A}$ after $q_s$ iterations, check that $H_1$-$coin_i = H_2$-$coin_i = H_4$-$coin_i = 1$ holds. If not, abort. Otherwise, check that the following conditions hold.

1. $\sigma_n^*$ is a valid signature on $\{m_j^*\}_{j=1,\cdots,n}$ in $\psi_n^*$ under $\{pk_j^*\}_{j=1,\cdots,n}$.

2. Exactly one $ID_{i^*}^*$ who has never been queried for both partial-private-key-extract and secret-value-extract exists.

3. Each $ID_i^*$ in $\{ID_j^*\}_{j=1,\cdots,n}$ does not appear more than once in $\psi_n^*$.

4. For $ID_{i^*}^*$, $m_{i^*}^* \notin \{m_{i^*}^{(1)}, \cdots, m_{i^*}^{(q_s)}\}$ or $\psi_{i^*}^* \notin \{\psi_{i^*}^{(1)}, \cdots, \psi_{i^*}^{(q_s)}\}$ holds.

Condition 4 described above means that either case 1 that $m_{i^*}^* \notin \{m_{i^*}^{(1)}, \cdots, m_{i^*}^{(q_s)}\}$ or case 2 that $m_{i^*}^* \in \{m_{i^*}^{(1)}, \cdots, m_{i^*}^{(q_s)}\} \wedge \psi_{i^*}^* \notin \{\psi_{i^*}^{(1)}, \cdots, \psi_{i^*}^{(q_s)}\}$ holds.

Here, the forgery can be written as $S^* = V^r \prod_{i=1}^n \left(g_{i,0}^a (g_{i,1}^a)^{c_i}\right) \prod_{i=1}^n W_i^{t_i}$, $R^* = g^r$ since this is a valid signature. The forgery belongs to either case 1 or case 2 described below from the condition 4, and $\mathscr{B}$ can extract the solution of CDH problem as follows:

case 1 ) $m_{i^*}^* \notin \{m_{i^*}^{(1)}, \cdots, m_{i^*}^{(q_s)}\}$ holds: $\mathscr{B}$ can extract $g'^a$ as follows:

$$g'^a = \left( \frac{\frac{S^*}{(R^*)^\beta \left(\prod_{j=1 \wedge j \neq i^*}^n T_i^\gamma\right)}}{\left(\prod_{j=1}^n (g^a)^{\alpha_{i,0}} (g^a)^{\alpha_{i,1} c_i}\right)} \right)^{\frac{1}{\alpha'_{i,0} + \alpha'_{i,1} c_i}} \tag{24}$$

case 2) $m_{i^*}^* \in \{m_{i^*}^{(1)}, \cdots, m_{i^*}^{(q_s)}\}$ but $\psi_{i^*}^* \notin \{\psi_{i^*}^{(1)}, \cdots, \psi_{i^*}^{(q_s)}\}$ holds: $\mathscr{B}$ can extract $g'^a$ as follows:

$$g'^a = \left( \frac{\frac{S^*}{(R^*)^\beta \left(\prod_{j=1 \wedge j \neq i^*}^n T_i^\gamma\right)}}{\left(\prod_{j=1}^n (g^a)^{\alpha_{i,0}} (g^a)^{\alpha_{i,1} c_i}\right) (g^a)^{\alpha_{i,0}}} \right)^{1/\alpha'_{i,0}} \tag{25}$$

Since $\mathscr{B}$ knows all the values, $\mathscr{B}$ can compute the above equation. The probability $\varepsilon'$ that $\mathscr{B}$ solves can be obtained as follows:

$$\varepsilon' \;=\; \Pr[forge \wedge \overline{abort} \wedge \overline{collide}] = \Pr[\overline{abort}] \cdot \left(\Pr[forge|\overline{abort}] - \Pr[collide|\overline{abort}]\right),$$

where $forge$ means an event that $\mathscr{A}$ succeeds in breaking the scheme, $collide$ means an event that $\mathscr{A}$ outputs $(\{m_j^*\}_{j,\cdots,n}, \psi_n^*, \sigma_n^*)$ such that it has previously been queried to *Signing* oracle and $abort$ means an event that $\mathscr{B}$ aborts the simulation with $\mathscr{A}$. $\Pr[forge|\overline{abort}] = \varepsilon$ holds from definition of the adversary and, from birthday paradox, $\Pr[collide|\overline{abort}]$ can be obtained as follows:

$$\Pr[collide|\overline{abort}] = \frac{q_{sig}(q_{sig}-1)}{2p} \tag{26}$$

In addition, $\Pr[\overline{abort}]$ can be written as follows:

$$\Pr[\overline{abort}] \;=\; \Pr[\overline{abort_p} \wedge \overline{abort_{h_4}} \wedge \overline{abort_{sig}} \wedge \overline{after}], \tag{27}$$

where $abort_p$ means the event that $\mathscr{B}$ aborts the simulation with $\mathscr{A}$ for partial-private-key-extract query. Similarly, We denote by $abort_{h_4}$ an event for $H_4$ query and by $abort_{sig}$ one for signing query. Here $abort_x$ means the event that $\mathscr{B}$ aborts the simulation with $\mathscr{A}$ during the $x$-query, where $x \in \{p, h_4, sig\}$ and each $p, h_4, sig$ stands for partial-private-key-extract, $H_4$ and signing, respectively. $after$ means that $\mathscr{B}$ aborts after $\mathscr{A}$ output the forgery. Each event can be written as follows:

$$\Pr[\overline{abort_p}] \;=\; (1-\delta)^{q_p}, \tag{28}$$

$$\Pr[\overline{abort_{h_4}}] \;=\; (1-\delta)^{q_{h_1}+q_{h_2}}, \tag{29}$$

$$\Pr[\overline{abort_{sig}}] \;=\; (1-\delta)^{(q_{h_1}+q_{h_2}+q_{h_4})q_{sig}}, \tag{30}$$

$$\Pr[\overline{after}] \;=\; \delta^3 \left(1+\frac{1}{n}\right), \tag{31}$$

where $\delta$ is a probability that $\mathscr{B}$ tosses 1 for the coin tosses. To complete the proof, we give the maximum value for $\delta$. Let $f(\delta)$ be the following function.

$$\begin{aligned} f(\delta) &= (1-\delta)^{q_p}(1-\delta)^{q_{h_1}+q_{h_2}}(1-\delta)^{(q_{h_1}+q_{h_2}+q_{h_4})q_{sig}}\delta^3 \\ &= (1-\delta)^{q_p+q_{h_1}+q_{h_2}+(q_{h_1}+q_{h_2}+q_{h_4})q_{sig}}\delta^3. \end{aligned} \tag{32}$$

To be easily written, we denote $a = q_p + q_{h_1} + q_{h_2} + (q_{h_1} + q_{h_2} + q_{h_4})q_{sig}$. From the derived function, $f$ is maximized at $\delta_{max} = \frac{3}{a}$. Here, $f(\delta_{max})$ can be written as follows:

$$f(\delta_{max}) = \frac{27}{a^3}\left(1-\frac{3}{a}\right)^a. \tag{33}$$

From definition of base of natural logarithm, we can compute as follows:

$$\lim_{a\to\infty}\left(1-\frac{3}{a}\right)^a = \frac{1}{e^3}, \tag{34}$$

$$\begin{aligned} \therefore \varepsilon' &= \left(\varepsilon - \frac{q_{sig}(q_{sig}-1)}{2p}\right) f_1(\delta_{opt}) \\ &= \left(\varepsilon - \frac{q_{sig}(q_{sig}-1)}{2p}\right)\cdot\frac{27}{a^3}\cdot\frac{1}{e^3}. \end{aligned} \tag{35}$$

The execution time of $\mathscr{B}$ is the execution time of $\mathscr{A}$ plus the computation time for $q_c$ create-user queries, $q_r$ public-key-replace queries, $q_p$ partial-private-key-extract queries, $q_s$ secret-value-extract queries, $q_{sig}$ signing queries, random oracle queries for each hash function and the computational time for the final step. Therefore,

$$t' = t + \mathscr{O}(q_{sig} + n(q_c + q_r + q_p + q_s + q_{h_1} + q_{h_2} + q_{h_3} + q_{h_4})) + \Psi, \tag{36}$$

where $\Psi$ is a computational time in the final step.

## B Proof of Theorem 9

This proof is based on the security proof in paper [31], and we also define a probability $\delta$ to set 1 for tossing a coin. To complete the proof, we finally determine a concrete value of $\delta$.

In this proof, we assume that $\mathscr{B}$ executes $H_1$-query and $H_2$-query before executing $H_3$-query and $H_4$-query, $H_1$-query before executing the create-user query and each random oracle query before executing the signing oracle query.

Given a CDH challenge value $(g, g^a, g')$, $\mathscr{B}$ who tries to solve CDH problem generates a pairing parameter $(p, \mathbb{G}, \mathbb{G}_T, \mathrm{e})$ and $g \in \mathbb{G}$. Then $\mathscr{B}$ generates $b \leftarrow \mathbb{Z}_p^*$ as $msk$, and sets $mpk = g^b$ and a certification list $\mathscr{L} = \emptyset$. Then $\mathscr{B}$ sets $ID$-list $[\cdot,\cdot,\cdot]$ $H_1$-list $[\cdot,\cdot,\cdot,\cdot,\cdot]$, $H_2$-list $[\cdot,\cdot,\cdot]$, $H_3$-list $[\cdot,\cdot,\cdot,\cdot,\cdot,\cdot]$ and $H_4$-list $[\cdot,\cdot,\cdot,\cdot,\cdot]$ as empty, and run $\mathscr{A}$ with $g, b, g^b$ as input.

$H_1$**-query** Given $ID_i$ generated by $\mathscr{A}$, check that $H_1$-list includes $ID_i$. If so, return $H_1(ID_i, j)$ from $H_1$-list, where $j = 0,1$. Otherwise, toss a coin $ID\text{-}coin_i \leftarrow \{0,1\}$ with probability $\delta$. and generate $\alpha_{i,0}, \alpha_{i,1} \leftarrow \mathbb{Z}_p$. If $ID\text{-}coin_i = 0$, set $H_1(ID_i, j) = g^{\alpha_{i,j}}$ for $j = 0,1$. Otherwise, set $H_1(ID_i, j) = (g^a)^{\alpha_{i,j}}$. Register $(ID_i, ID\text{-}coin_i, \cdot)$ on $ID$-list and $(ID_i, \alpha_{i,0}, \alpha_{i,1}, \alpha'_{i,0}, \alpha'_{i,1})$ on $H_1$-list, and return $H_1(ID_i, j)$, $j = 0,1$.

$H_2$**-query** This execution is exactly the same as Game 1.

$H_3$**-query** Given $s \parallel L_i$ generated by $\mathscr{A}$, check that $H_3$-*list* includes $s \parallel L_i$. If so, return $H_3(s \parallel L_i)$ from $H_3$-*list*. Otherwise, generate $\gamma \leftarrow \mathbb{Z}_p^*$ and toss a coin $H_3$-$coin_l \leftarrow \{0,1\}$ with the probability $\delta$. If $H_3$-$coin_l = 0$, set $H_3(s \parallel L_i) = g^{\gamma}$. Otherwise, $H_3(s \parallel L_i) = (g \cdot g')^{\gamma}$. Register $(s, ID_i, m_i, L_i, H_3\text{-}coin_l, \gamma)$ on $H_3$-list and return $H_3(s \parallel L_i)$.

$H_4$**-query** Given $s \parallel L_i$ generated by $\mathscr{A}$, check that $H_4$-*list* includes $s \parallel L_i$. If so, return $H_4(s \parallel L_i)$ from $H_4$-*list*. Otherwise, check that $L_i \neq L_i'$ exists with $(m_i, ID_i) = (m_i', ID_i')$. If not, set $H_4(s \parallel L_i) = -d_{(i,k,l)}$ such that $\gamma t_i + (\alpha_{i,0} + \alpha_{i,1} d_{(i,k,l)}) b = 0$. Otherwise, generate $d_{(i,k,l)} \leftarrow \mathbb{Z}_p^*$ and set $H_4(s \parallel L_i) = d_{(i,k,l)}$. Register $(s, ID_i, m_i, L_i, d_{(i,k,l)})$ on $H_4$-list and return $H_4(s \parallel L_i)$.

**Create-User** Given $ID_i$ generated by $\mathscr{A}$, check that $\mathscr{L}$ includes $ID_i$. If so, return $(ID_i, T_i)$ from $\mathscr{L}$. Otherwise, retrieve $ID$-$coin_i$ from $ID$-list and $H_i(ID_i, j)$ for $j = 0, 1$ from $H_1$-list as $g_{i,j}$, and generate $t_i \leftarrow \mathbb{Z}_p$. If $ID$-$coin_i = 0$, set $g_{i,j}^b$ as a partial private key and $T_i = g^{t_i}$. Otherwise, set $g_{i,j}^b$ and $T_i = (g^a)^{t_i}$. Register $(ID_i, T_i)$ in $\mathscr{L}$ and re-register $(ID_i, ID\text{-}coin_i, t_i)$ in $ID$-list. Return $H_1(ID_i, j), T_i$ as $ID_i$'s public key $pk_i$.

**Secret-Value-Extract** Given $ID_i$ generated by $\mathscr{A}$, check that $\mathscr{L}$ include $ID_i$. If not, nothing will be output. Otherwise, check that $ID$-$coin_i = 1$ holds. If so, abort. Otherwise, return $t_i$.

**Signing** Given a signing query $(\{m_j\}_{j=1,\cdots,i}, \{ID_j\}_{j=1,\cdots,i}, \psi_i, \sigma_{i-1}, s)$ generated by $\mathscr{A}$, check that $ID$-$coin_i$, $H_2$-$coin_k$ and $H_3$-$coin_l$ in the query with each list. If $ID$-$coin_i = H_2$-$coin_k = H_3$-$coin_l = 1$, abort. Otherwise, generate a random number $r \leftarrow \mathbb{Z}_p^*$ and generate a signature as follows. In the case that $H_3$-$coin_l = 0$, compute as follows:

$$S_i = V^r g_{i,0}^b \left(g_{i,1}^b\right)^{c_i} (T_i)^{\gamma} \cdot S_{i-1}, \tag{37}$$

$$R_i = g^r \cdot R_{i-1}, \tag{38}$$

where $V, \gamma$ and $c_i$ are retrieved from $H_2$-*list*, $H_3$-*list* and $H_4$-*list*. These values become a valid signature on $\{m_j\}_{j=1,\cdots,i}$ in $\psi_i$ for $\{ID_j\}_{j=1,\cdots,i}$ since the following equation holds:

$$\begin{aligned} S_i &= V^r g_{i,0}^b \left(g_{i,1}^b\right)^{c_i} (T_i)^{\gamma} \cdot S_{i-1} = V^r g_{i,0}^b \left(g_{i,1}^b\right)^{c_i} (g^{x_i})^{\gamma} \cdot S_{i-1} \\ &= V^r g_{i,0}^b \left(g_{i,1}^b\right)^{c_i} (W_i)^{x_i} \cdot S_{i-1}, \end{aligned} \tag{39}$$

where $x_i$ is a secret key corresponding to a public key $T_i$. In the case that $H_3$-$coin_i = 1 \wedge H_2$-$coin_i = 0$, compute as follows:

$$S_i = \left(g'^{\beta}\right)^r g_{i,0}^b \left(g_{i,1}^b\right)^{c_i} (T_i)^{\gamma} \cdot S_{i-1}, \tag{40}$$

$$R_i = g^r (T_i)^{-\frac{\gamma}{\beta}} \cdot R_{i-1}, \tag{41}$$

These values also become a valid signature on $\{m_j\}_{j=1,\cdots,i}$ in $\psi_i$ for $\{ID_j\}_{j=1,\cdots,i}$ since the following equation holds:

$$\begin{aligned} S_i &= \left(g'^{\beta}\right)^r g_{i,0}^b \left(g_{i,1}^b\right)^{c_i} (T_i)^{\gamma} \cdot S_{i-1} = \left(g'^{\beta}\right)^r g_{i,0}^b \left(g_{i,1}^b\right)^{c_i} (g^{x_i})^{\gamma} (g')^{x_i\gamma - x_i\gamma} \cdot S_{i-1} \\ &= \left(g'^{\beta}\right)^r g_{i,0}^b \left(g_{i,1}^b\right)^{c_i} (g^{x_i} g'^{x_i})^{\gamma} (g')^{-x_i\gamma} \cdot S_{i-1} = \left(g'^{\beta}\right)^{r - \frac{x_i\gamma}{\beta}} g_{i,0}^b \left(g_{i,1}^b\right)^{c_i} (g \cdot g')^{x_i\gamma} \cdot S_{i-1} \\ &= \left(g'^{\beta}\right)^{r - \frac{x_i\gamma}{\beta}} g_{i,0}^b \left(g_{i,1}^b\right)^{c_i} (W_i)^{x_i} \cdot S_{i-1}, \end{aligned} \tag{42}$$

$$R_i = g^r \cdot (g^{x_i})^{-\frac{\gamma}{\beta}} \cdot R_{i-1} = g^{r - \frac{x_i\gamma}{\beta}} \cdot R_{i-1}. \tag{43}$$

Here, we write $r' = r - \frac{\gamma x_i}{\beta}$. Then, the following equations can be written:

$$S_i = (V)^{r'} g_{i,0}^{b} \left(g_{i,1}^{b}\right)^{c_i} (W_i)^{x_i} \cdot S_{i-1}, \tag{44}$$

$$R_i = g^{r'} \cdot R_{i-1}. \tag{45}$$

**Output** Given a forgery $(\{ID_j^*\}_{j=1,\cdots,n}$ by $\mathscr{A}$, $\{m_j^*\}_{j=1,\cdots,n}, \psi_n^*, \sigma_n^*)$ output by $\mathscr{A}$ after $q_s$ iterations, check that $ID\text{-}coin_i = H_2\text{-}coin_i = H_3\text{-}coin_i = 1$ holds. If not, abort. Otherwise, check that the following conditions hold.

1. $\sigma_n^*$ is a valid signature on $\{m_j^*\}_{j=1,\cdots,n}$ in $\psi_n^*$ under $\{pk_j^*\}_{j=1,\cdots,n}$.
2. Exactly one $ID_{i^*}^*$ who has never been queried for secret-value-extract exists and replace-public-key.
3. Each $ID_i^*$ in $\{ID_j^*\}_{j=1,\cdots,n}$ does not appear more than once in $\psi_n^*$.
4. For $ID_{i^*}^*$, $m_{i^*}^* \notin \{m_{i^*}^{(1)}, \cdots, m_{i^*}^{(q_s)}\}$ or $\psi_{i^*}^* \notin \{\psi_{i^*}^{(1)}, \cdots, \psi_{i^*}^{(q_s)}\}$ holds.

Similarly with Game 1, $\mathscr{B}$ can extract the solution of CDH problem from the forgery as follows:

case 1) $\mathscr{B}$ can extract $g'^a$ as follows:

$$g'^a = \left( \frac{\frac{S^*}{(R^*)^{\beta}\left(\prod_{j=1 \wedge j \neq i^*}^{n} T_j^{\gamma}\right)}}{(g^a)^{t_{i^*}\gamma}\left(\prod_{j=1}^{n} g_{j,0}^{b}(g_{j,1}^{b})^{c_i}\right)} \right)^{\frac{1}{t_{j^*}\gamma}}. \tag{46}$$

case 2) $\mathscr{B}$ can extract $g'^a$ as follows:

$$g'^a = \left( \frac{\frac{S^*}{(R^*)^{\beta}\left(\prod_{j=1 \wedge j \neq i^*}^{n} T_i^{\gamma}\right)}}{\left(\prod_{i=1}^{n} g_{i,0}^{b}(g_{i,1}^{b})^{c_i}\right)} \right)^{1/t_{i^*}\gamma}. \tag{47}$$

Here, the above equation can be written as follows:

$$\begin{aligned} g'^a &= \left( \frac{W_{i^*}^{at_{i^*}} \prod_{j=1}^{n} g_{j,0}^{b} g_{j,1}^{bc_j}}{\prod_{j=1 \wedge j \neq i^*}^{n} g_{j,0}^{b} g_{j,1}^{bc_j}} \right)^{1/t_{i^*}\gamma} = \left( (g \cdot g')^{at_{i^*}\gamma} (g^a)^{(\alpha_{i^*,0} + \alpha_{i^*,1} d_{i^*kl})b} \right)^{1/t_{i^*}\gamma} \\ &= \left( (g')^{at_{i^*}\gamma} \right)^{1/t_{i^*}\gamma}, \end{aligned} \tag{48}$$

because $\gamma t_{i^*} + (\alpha_{i^*,0} + \alpha_{i^*,1} d_{i^*kl})b = 0$ from $H_4$-list.

Since $\mathscr{B}$ knows all the values, $\mathscr{B}$ can compute the above equation. Similarly with Game 1, we can compute the maximum value for $f(\delta)$ and the computational time for $\mathscr{B}$.

$$\varepsilon' = \Pr[forge \wedge \overline{abort} \wedge \overline{collide}] = \Pr[\overline{abort}] \cdot \left(\Pr[forge|\overline{abort}] - \Pr[collide|\overline{abort}]\right).$$

Here, $\Pr[forge|\overline{abort}] = \varepsilon$ and $\Pr[collide|\overline{abort}] = \frac{q_{sig}(q_{sig}-1)}{2p}$ hold similarly with Game 1. In addition, $\Pr[\overline{abort}]$ can be written as follows:

$$\Pr[\overline{abort}] = \Pr[\overline{abort_s} \wedge \overline{abort_{h_3}} \wedge \overline{abort_{sig}} \wedge \overline{after}], \tag{49}$$

where $abort_s$ means the event that $\mathscr{B}$ aborts the simulation with $\mathscr{A}$ for secret-value-extract query. Each event can be written as follows:

$$\begin{aligned} \Pr[\overline{abort_s}] &= (1-\delta)^{q_s}, & (50)\\ \Pr[\overline{abort_{sig}}] &= (1-\delta)^{(q_{h_1}+q_{h_2}+q_{h_3})q_{sig}}, & (51)\\ \Pr[\overline{after}] &= \delta^3\left(1+\frac{1}{n}\right), & (52) \end{aligned}$$

where $\delta$ is a probability that $\mathscr{B}$ tosses 1 for its coin tosses. Similarly with Game 1, we give the maximum value for $\delta$. We define $f(\delta)$ as the following function.

$$\begin{aligned} f(\delta) &= (1-\delta)^{q_s}(1-\delta)^{(q_{h_1}+q_{h_2}+q_{h_3})q_{sig}}\delta^3 \\ &= (1-\delta)^{q_s+(q_{h_1}+q_{h_2}+q_{h_s})q_{sig}}\delta^3. & (53) \end{aligned}$$

Here, we denote $a = q_s + (q_{h_1} + q_{h_2} + q_{h_4})q_{sig}$. From the derived function, $f$ is maximized at $\delta_{max} = \frac{3}{a}$. Therefore, similarly with Game 1, the following equation can be obtained.

$$\begin{aligned} \varepsilon' &= \left(\varepsilon - \frac{q_{sig}(q_{sig}-1)}{2p}\right) f_1(\delta_{max}) \\ &= \left(\varepsilon - \frac{q_{sig}(q_{sig}-1)}{2p}\right)\cdot\frac{3}{a^3}\cdot\frac{1}{e^3}. & (54) \end{aligned}$$

The execution time of $\mathscr{B}$ can be also obtained similarly with Game 1. The execution time is the execution time of $\mathscr{A}$ plus the computation time for $q_c$ create-user queries, $q_r$ public-key-replace queries, $q_s$ secret-value-extract queries, $q_{sig}$ signing queries, random oracle queries for each hash function and the computational time for the final step. Therefore,

$$t' = t + \mathscr{O}(q_{sig} + n(q_c + q_r + q_s + q_{h_1} + q_{h_2} + q_{h_3} + q_{h_4})) + \Psi, \quad (55)$$

where $\Psi$ is a computational time in the final step.