Batch Verifications with ID-based Signatures

HyoJin Yoon[1⋆], Jung Hee Cheon[1⋆] and Yongdae Kim[2⋆⋆]

[1] Department of Mathematical Sciences,
Seoul National University, Korea
{jhcheon,jin25}@math.snu.ac.kr
[2] Department of Computer Science
University of Minnesota - Twin Cities, USA
kyd@cs.umn.edu

Abstract. An identity (ID)-based signature scheme allows any pair of users to verify each other's signatures without exchanging public key certificates. With the advent of Bilinear maps, several ID-based signatures based on the discrete logarithm problem have been proposed. While these signatures have an advantage in the fact that the system secret can be shared by several parties using a threshold scheme (thereby overcoming the security problem of RSA-based ID-based signature schemes), they all share the same efficiency disadvantage. To overcome this, some schemes have focused on finding ways to verify multiple signatures at the same time (i.e. the batch verification problem). While they had some success in improving efficiency of verification, each had a slightly diversified definition of *batch verification*. In this paper, we propose a taxonomy of batch verification against which we analyze security of well-known ID-based signature schemes. We also propose a new ID-based signature scheme that allows for all types of multiple signature batch verification, and prove its security in random oracle model.

Key words : ID-based signatures, Batch verifications

1 Introduction

In 1984, Shamir proposed a new model for public key cryptography, called identity (ID)-based encryption and signature schemes. The goal was to simplify key management procedures of certificate-based public key infrastructures (PKIs) [27]. Since then, several ID-based encryption and signature schemes. based on integer factorization problem, have been proposed [9, 29, 30, 21]. While these ID-based signatures have improved key management and key recovery, their disadvantage lies in the fact that the signer's key is shared with the private key generator [13, 10]. This problem can be alleviated using signatures based

⋆ The first and second authors were supported in part by Korea Telecom.
⋆⋆ The third author is supported in part by DTC Intelligent Storage Consortium at the University of Minnesota. Part of this work was done while the third author was visiting Seoul National University in 2003.

on the discrete logarithm problem (DLP) instead, since in this case the secret key can be shared by several parties using a threshold scheme. Several ID-based signatures with these properties that use pairings in elliptic curves have been proposed [14, 25, 8].

In spite of several advantages of ID-based signatures schemes based on pairings, they suffer from an efficiency problem that puts restrictions on their use in applications: Their signature verifications are between ten and two hundred times slower than those of DSS or RSA [1]. This problem may be critical in some applications such as electronic commerce and banking service, in which one server may have to verify many signatures simultaneously. To improve the efficiency of performance for multiple signature verification, many researchers have studied so called batch verification.

Even so, each proposed approach has a different definition of batch verification. We classify multiple signatures (i.e. input of batch verification) into the following three types, according to the number of signers and messages:

Type 1 multiple signatures on a single message generated by multiple signers
Type 2 multiple signatures on multiple messages generated by a single signer
Type 3 multiple signatures on multiple messages generated by multiple signers, where each message is signed by a distinct user

Type 1 signature was traditionally classified as multisignature and has been studied for a long time [16, 23, 24, 20, 6]. Due to its simplicity, it allows for very efficient batch verification. *Type 2* batch verification proposals centered around batch RSA [11, 2] and have been a topic of research since late 80's. Compression of multiple RSA signatures of type 2 into one signature is also called condensed RSA [19]. More precisely, our discussion deals with different notion of batch verification, called *screening* [2]. That is, we only want to determine whether *the signer has at some point authenticated the text* rather than verifying if each string provided is the valid signature corresponding to the message. Recently, Boneh *et al.* proposed aggregate signatures (BGLS scheme) using bilinear maps, in which multiple signatures are aggregated into a single signature [3]. They allow for batch verification of *type 3*, but the efficiency gain is almost half of usual verifications. We note that there have been many efforts that aim at speeding up simultaneous verifications of modular exponentiations for DSA signatures [22, 18, 2, 7]. These approaches are *independent* of specific signature schemes, but the efficiency gain over the sum of individual verifications is limited. On the other hand, our approach can give significant improvement in efficiency.

Our Contributions In this paper, we discuss batch verifications of ID-based signatures according to the above taxonomy. **(1)** We discuss security of batch verification of type 2 in the Cha-Cheon scheme. We provide a loose security reduction of batch verification of type 2 in Cha-Cheon scheme to the *computational Diffie-Hellman problem* (CDHP). It is the same as in the Hess scheme [14]. **(2)** We show that previous signature schemes are not secure in batch verification of Type 1 or 3. **(3)** We propose a new ID-based signature scheme that is secure

in batch verification of Type 1 and 3 and provide security proof under random oracle model.

Organizations The rest of the paper is organized as follows: In Section 2, we introduce hard problems which our scheme relies upon. In Section 3, we analyze previously proposed ID-based signatures in batch verification of Type 2. We also discuss why those ID-based signature schemes fail to provide secure batch verification of Type 3 and Type 1. In Section 4, we propose a new ID-based signature scheme admitting secure batch verification of Type 3 and Type 1, provide proof of security and discuss the batch verification of each type. We conclude in Section 5.

2 Preliminary

2.1 Bilinear Maps

Consider an additive cyclic group G of prime order ℓ and a cyclic multiplicative group V. Let $e\colon G \times G \to V$ be a map which satisfies the following properties.

1. **Bilinear** For any $aP, bP \in G$, $e(aP, bP) = e(P, P)^{ab}$.
2. **Non-degenerate** If $e(P, Q) = 1_V$ for all P (or Q) in G, then Q (or P) is the identity of G, respectively.
3. **Efficient** There exists an efficient algorithm to compute the map.

We call such a bilinear map as an admissible bilinear pairing.

The Weil pairing and Tate pairing in elliptic curve give good implementations of the admissible bilinear pairing. Let E be an elliptic curve over $\mathbb{F}_q$ where $q = p^n$ and p is a prime. For a prime ℓ and an ℓ torsion subgroup $E[\ell]$ of E, we define a Weil pairing $e_W : E[\ell] \times E[\ell] \to \mathbb{F}^*_{q^\alpha}$ for suitable α. Now let $G = E(\mathbb{F}_q)[\ell]$ and define a map $e\colon G \times G \to \mathbb{F}^*_{q^\alpha}$, where $e(P, Q) = e_W(P, \phi(Q))$ and ϕ is an automorphism over G. Then e is an efficiently computable non-degenerate bilinear map. The Tate pairing has similar properties and is more efficient than the Weil pairing. For the details, refer to [4].

2.2 Some Problems

Let G be a cyclic group of prime order ℓ and P a generator of G.

1. The decisional Diffie-Hellman Problem (DDHP) is to decide whether $c = ab$ in $\mathbb{Z}/\ell\mathbb{Z}$ for given $P, aP, bP, cP \in G$. If so, (P, aP, bP, cP) is called a valid Diffie-Hellman (DH) tuple.
2. The computation Diffie-Hellman Problem (CDHP) is to compute abP for given $P, aP, bP \in G$.

Now we define a gap Diffie-Hellman group.

Definition 1 *A group G is a gap Diffie-Hellman group if the DDHP in G can be efficiently computable and there exists no algorithm which can solve the CDHP in G with non-negligible probability within polynomial time.*

If we have an admissible bilinear pairing e in G, we can solve the DDHP in G efficiently as follows:

$$(P, aP, bP, cP) \text{ is a valid DH tuple} \Leftrightarrow e(aP, bP) = e(P, cP).$$

Hence an elliptic curve becomes an instance of a gap Diffie-Hellman group if the Weil (or the Tate) pairing is efficiently computable and the CDHP is sufficiently hard on the curve.

From now on, we assume that G is a gap Diffie-Hellman group generated by P, whose order is a large prime ℓ and all schemes are performed in the group G if not special remarks. To implement the a gap Diffie-Hellman group we consider G as a subset of elliptic curve as above with an admissible pairing e originated from the Weil pairing or Tate pairing.

2.3 ID-based Signature Schemes and Attack Models for Batch Verifications

An ID-based signature scheme consists of four algorithms: *Setup, Extract, Signing* and *Verification*.

Setup A key generation center (KGC) sets the system's secret key K_s that is called the master key and the system parameters Param.

Extract For each identity ID, KGC generates the secret key D_{ID} corresponding to ID using K_s and Param.

Signing A user with ID produces a signature (ID, m, σ) on a message m using her secret key D_{ID} and Param.

Verification Given the signature (ID, m, σ), a verifier checks the validity of the σ using Param.

In batch verification, we replace the **Verification** process by the following process:

Batch Verification Given multiple signatures $\sigma_1, \cdots, \sigma_k$ on messages $m_1, \cdots, m_k$ and corresponding identities $\mathrm{ID}_1, \cdots, \mathrm{ID}_k$, a verifier checks the validity of all signatures at once.

In the batch verification, if $m_1 = \cdots = m_k$ then we call this the batch verification of (multiple signatures) of Type 1. If $\mathrm{ID}_1 = \cdots = \mathrm{ID}_k$ then we call this the batch verification of Type 2. If each message is signed by distinct ID's then we call this the batch verification of Type 3.

We formalize the attack model for batch verification of Type 1, 2 and 3 in the general ID-based signature scheme. We call a forger $\mathcal{F}$ a k-batch forger of Type i, where i=1, 2, 3, when $\mathcal{F}$ executes the following game. Note that $\mathcal{F}$ performs an existential forgery under the adaptively chosen message and ID attack.

Setup A k-batch forger $\mathcal{F}$ is given public system parameters.
Queries $\mathcal{F}$ can access the hash, **Extract** and **Signing** oracle. $\mathcal{F}$ obtains the hash values of his queries, the secret keys of his chosen ID's and the signatures of his chosen ID's and messages.
Outputs Finally, $\mathcal{F}$ outputs $\mathrm{ID}_1, \cdots, \mathrm{ID}_n$ and message $m_1, \cdots, m_n$ and corresponding signatures $\sigma_1, \cdots, \sigma_n$ of Type i where $n \leq k$ and $i = 1,\ 2,\ 3$.

$\mathcal{F}$ wins if the outputs pass the batch verification process of each type within polynomial time bound with non-negligible probability and there exists one index i such that the ID_i has not been queried to the **Extract** oracle and the message m_i corresponding to ID_i has not been asked to the **Signing** oracle.

3 Batch Verifications in ID-based Signatures

In this section, we discuss the security of previous ID-based signature schemes.

3.1 Batch Verifications of Type 2 in the Cha-Cheon Scheme

The Cha-Cheon ID-based signature scheme consists of four algorithms: *Setup, Extract, Signing* and *Verification*.

Setup Given a gap Diffie-Hellman group G with an admissible pairing e and its generator P, pick a random $s \in \mathbb{Z}/\ell\mathbb{Z}$ and set $P_{pub} = sP$. Choose two hash functions $H_1 : \{0,1\}^* \times G \to (\mathbb{Z}/\ell\mathbb{Z})^*$ and $H_2 : \{0,1\}^* \to G^*$. The system parameter is (P, P_{pub}, H_1, H_2). The master key is s.
Extract Given an identity ID, the algorithm computes $Q_{\mathrm{ID}} = H_2(\mathrm{ID})$ and $D_{\mathrm{ID}} = sH_2(\mathrm{ID})$ and outputs D_{ID} as a private key of the identity ID.
Signing Given a secret key D_{ID} and a message m, pick a random number $r \in \mathbb{Z}/\ell\mathbb{Z}$ and output a signature $\sigma = (m, U, h, V)$ where $U = rQ_{\mathrm{ID}}$, $h = H_1(m, U)$, and $V = (r + h)D_{\mathrm{ID}}$.
Verification Given a signature $\sigma = (U, V)$ of a message m for an identity ID, compute $h = H_1(m, U)$. The signature is accepted if and only if $e(P, V) = e(P_{pub}, U + hQ_{\mathrm{ID}})$.

Let $\sigma_i = (m_i, U_i, h_i, V_i)$ be the signatures using the Cha-Cheon scheme signed by a single user with ID on distinct k-messages m_i, $U_i = r_iQ_{\mathrm{ID}}$, $V_i = (r_i + h_i)D_{\mathrm{ID}}$ and $h_i = H_1(m_i, U_i)$ where $i = 1, 2, \cdots k$, $Q_{\mathrm{ID}} = H_2(\mathrm{ID})$ and D_{ID} is a secret key of user. Then we can verify all k-signatures at once as follows:

- Compute $Q_{\mathrm{ID}} = H_2(\mathrm{ID})$ and $h_i = H_1(m_i, U_i)$ for all $i = 1, \cdots, k$.
- Check whether $e(P, \sum_{i=1}^{k} V_i) = e\left(P_{pub}, \sum_{i=1}^{k} U_i + (\sum_{i=1}^{k} h_i)Q_{\mathrm{ID}}\right)$ or not.

We know that the Cha-Cheon scheme is secure in gap Diffie-Hellman group in random oracle model [8]. Now we analyze the security of batch verification of Type 2 in the Cha-Cheon scheme.

Theorem 1. *Let $\mathcal{F}_0$ be k-batch forger of Type 2 which performs an existential forgery under an adaptively chosen message and* ID *attack against the Cha-Cheon scheme within a time bound T_0 with probability ϵ_0 in random oracle model. The forger $\mathcal{F}_0$ can ask queries to the oracles H_1, H_2,* **Extract** *and* **Signing** *at most q_{H_1}, q_{H_2}, q_E, and q_S-times, respectively. And $V_{q_{H_1},k}$ denotes k times the number of k-permutations of q_{H_1} elements, that is, $V_{q_{H_1},k} = k \cdot q_{H_1}(q_{H_1} - 1) \cdots (q_{H_1} - k + 1)$. If $\epsilon_0 \geq (12V_{q_{H_1},k} + 6(q_{H_1} + k \cdot q_S)^2)q_{H_2}/(\ell - 1)$, then the CDHP can be solved with probability $\geq$ 1/9 and within running time $\leq 144823V_{q_{H_1},k}(1 + q_S)q_{H_2}T_0/\left(\epsilon_0\left(1 - \frac{1}{\ell}\right)\right)$.*

To prove the above theorem, we consider the properties of the ID-based scheme. While each secret key of user is chosen independently in the traditional public key system, all secret keys of users are mutually related in ID-based system. In fact, they are produced from one secret key of the whole system which is called the master key. Hence in ID-based setting it is reasonable to give not an specific ID but a system parameter to a forger. Using [8, Lemma 1], we can reduce the adaptively chosen ID attack to the *given* ID attack.

Now, consider the following lemma to reduce the security of batch verification of Type 2 in the Cha-Cheon scheme to the CDHP under the given ID attack model.

Lemma 1. *Let $\mathcal{F}$ be k-batch forger of Type 2 which performs an existential forgery under an adaptively chosen message and given* ID *attack against the Cha-Cheon scheme within a time bound T with probability ϵ in random oracle model. The forger $\mathcal{F}$ can ask queries to the oracles H_1, H_2,* **Extract** *and* **Signing** *at most q_{H_1}, q_{H_2}, q_E, and q_S-times, respectively. We assume that, within time bound T, $\mathcal{F}$ produces, with probability of success $\epsilon \geq \frac{12V_{q_{H_1},k}+6(q_H+k\cdot q_S)^2}{\ell}$, multiple signatures $\sigma = (m_i, U_i, h_i, V_i)$, $i = 1, 2, \cdots, n$ and $n \leq k$, which pass the batch verification. Then, there is another probabilistic polynomial time Turing machine which has control over the machine obtained from $\mathcal{F}$ by simulation, and which produces another multiple signatures $\sigma'_i = (m'_i, U_i, h'_i, V'_i)$, $i = 1, 2, \cdots, n$ such that $h_j \neq h'_j$, for some $j \in \{1, \cdots, n\}$ and $h_i = h'_i$ for all $i = 1, \cdots, n$ such that $i \neq j$ within time $T' = \frac{144823V_{q_{H_1},k}(1+q_S)T}{\epsilon}$.*

The Lemma 1 can be proved using the similar method with [15, Theorem 2] except the number of signatures in output $n \in \{1, \cdots, k\}$. In [15], they deal with the ring signature, so the number of signatures i.e. random parts R_i's are fixed as the number of users in the ring. But in the batch verification of Type 2, the number of signatures which is batch verified is not fixed, it is only less than k. So to fix the number of signatures during the oracle replay, we use a random variable tuple (ω, n, f) not (ω, f) when we apply the splitting lemma. Note that the number of signatures n is included in the fixed parts when we apply the splitting lemma. Thus we need k times the original $V_{q_{H_1},k}$ in [15, Theorem 2].

Proof (of theorem 1). Using the above Lemma 1, we can prove the theorem 1 in the given ID attack case. We construct an algorithm $\mathcal{C}$ to solve the CDHP

using the forger $\mathcal{F}$. We assume that P, aP, bP are given as the CDHP instances. The algorithm $\mathcal{C}$ simulates a real signer to get signatures which pass the batch verification of Type 2 from $\mathcal{F}$. If $\mathcal{C}$ does not fail this simulation, $\mathcal{C}$ gets multiple signatures what he wants and using the general oracle replaying technique, $\mathcal{C}$ can solve the CDHP. In Setup, the algorithm $\mathcal{C}$ fixes a target identity ID, and put $P_{pub} = aP$.

Note that ID-**Hash Query, Extract Query, Message-Hash Query**, and **Signing Query** are the same as the proof of [8, Lemma 2]. After the queries, if the simulation does not fail, the forger $\mathcal{F}$ outputs multiple signatures $\sigma_i = (m_i, U_i, h_i, V_i)$ for given ID, where $i = 1, 2, \cdots, n$ and $n \leq k$. Then $\mathcal{C}$ replays the oracles and obtains another multiple signatures $\sigma_i' = (m_i, U_i, h_i', V_i')$, $i = 1, 2, \cdots n$ using the Lemma 1. Let $V = \sum_{i=1}^{n} V_i$, $V' = \sum_{i=1}^{n} V_i'$. Since the signatures pass the batch verification of Type 2, $\mathcal{C}$ knows that the following two equations are satisfying:

$$e(P, V) = e\left(P_{pub}, \sum_{i=1}^{n} U_i + \left(\sum_{i=1}^{n} h_i\right) Q_{\mathrm{ID}}\right)$$
$$e(P, V') = e\left(P_{pub}, \sum_{i=1}^{n} U_i + \left(\sum_{i=1}^{n} h_i'\right) Q_{\mathrm{ID}}\right)$$

Thus from $V = \sum_{i=1}^{n}(r_i + h_i)D_{\mathrm{ID}}$ and $V' = \sum_{i=1}^{n}(r_i + h_i')D_{\mathrm{ID}}$, $\mathcal{C}$ can compute $V - V' = \sum_{i=1}^{n}(h_i - h_i')D_{\mathrm{ID}}$. Since there exist an $i \in \{1, \cdots n\}$ such that $h_i \neq h_i'$, $\mathcal{C}$ obtains $abP = D_{\mathrm{ID}} = (V - V')/\sum_{i=1}^{n}(h_i - h_i')$. The total running time is bounded by the running time of the Lemma [15]. Thus applying [8, Lemma 1], we obtain the result of theorem 1. □

Remark 2 *We may consider batch verification of Type 2 in the Hess scheme. In the original Hess scheme, we must compute a hash value and compare it with some value to verify. But a hash function does not have any homomorphic property, thus we cannot use directly the original Hess scheme for batch verification. Hence we slightly modify the signing and verification processes in the Hess scheme to apply the batch verification. Let the signature of a user with ID be* $\sigma = (\mathrm{ID}, m, R, h, V)$ *where* $h = H_1(m, U)$, $U = e(P, R)$, $V = hD_{\mathrm{ID}} + R$ *and* D_{ID} *is a secret key of user. Then the batch verification of Type 2 in the Hess scheme is possible and the security of them can be reduced to the CDHP similarly to the above theorem.*

The time complexity of the reduction is dominated by T_0 times k-th power of the number of H_1 hash queries over ϵ_0. That is, if k increases then the time complexity of security reduction increases exponentially. Thus the Theorem 1 gives a security proof of the batch verification in the Cha-Cheon scheme of Type 2 only when the of signature k is very small. It is the same for that of the Hess scheme.

3.2 Batch Verification of Type 3 in the Cha-Cheon Scheme

We also consider the batch verification in the Cha-Cheon [8] scheme of Type 3. However, it is not secure.

Let ID_1 be an identity of honest user $\mathcal{U}_1$ and ID_2 an identity of a 2-batch forger $\mathcal{F}$ of Type 3 in the Cha-Cheon scheme. We may assume that $\mathcal{F}$ can access to the **ID-hash** oracle and obtain $Q_1 - H_2(ID_1)$, $Q_2 = H_2(ID_2)$. Now $\mathcal{F}$ selects two random values $r_1, \widetilde{r_2}$ and messages m_1, m_2, compute $U_1 = r_1 Q_1$, $h_1 = H_1(m_1, U_1)$ and

$$U_2 = \widetilde{r_2} Q_2 - h_1 Q_1 - r_1 Q_1.$$

Finally, $\mathcal{F}$ computes $h_2 = H_2(m_2, U_2)$ and $V_1 = (r_2' + h_2)D_2$, $V_2 = r_2'' D_2$, where $r_2' + r_2'' = \widetilde{r_2}$, and outputs two signatures $\sigma_1 = (ID_1, m_1, U_1, h_1, V_1)$ and $\sigma_2 = (ID_2, m_2, U_2, h_2, V_2)$. Though $\mathcal{F}$ does not know the discrete log of U_2, r_2, these multiple signatures pass the batch verification of Type 3:

$$\begin{aligned} e(P_{pub}, U_1 + h_1 Q_1 + U_2 + h_2 Q_2) &= e(P, r_1 D_1 + h_1 D_1 + \widetilde{r_2} D_2 - h_1 D_1 - r_1 D_1 + h_2 D_2) \\ &= e(P, \widetilde{r_2} D_2 + h_2 D_2) \\ &= e(P, V_1 + V_2). \end{aligned}$$

That is, $\mathcal{F}$ pretends to generate signatures which pass the batch verification of Type 3 with the honest user U_1. Thus the batch verification of Type 3 in the Cha-Cheon scheme is not secure.

Remark 3 *In the case of the Hess scheme, because of the same reason with the previous subsection, we consider the modified Hess scheme. Similarly to the Cha-Cheon scheme, let* $U_2 = e(P_{pub}, -h_1 Q_1) \cdot e(P, \widetilde{R_2}) = e(P, -h_1 D_1 + \widetilde{R_2})$ *where a random point* R_2 *is the same role as* $U_2 = r_2 Q_{ID}$ *in the Cha-Cheon scheme, then the forged signatures generated by the forger alone pass the batch verification.*

In the Cha-Cheon and Hess scheme, a random part U is used as an input of the hash function H_1. Thus although all messages are same, the hash outputs are all distinct. So the batch verification of Type 1 in the Cha-Cheon scheme is the same as that of Type 3.

4 ID-based Signature Scheme Admitting Batch Verification of Type 3

4.1 New ID-based Signature Scheme

This scheme consists of four algorithms: *Setup, Extract, Signing* and *Verification*.

Setup Given a gap Diffie-Hellman group G with an admissible pairing e and its generator P, pick a random $s \in \mathbb{Z}/\ell\mathbb{Z}$ and set $P_{pub} = sP$. Choose two hash functions $H_1 : \{0,1\}^* \times G \to (\mathbb{Z}/\ell\mathbb{Z})^*$ and $H_2 : \{0,1\}^* \to G^*$. The system parameter is (P, P_{pub}, H_1, H_2). The master key is s.

Extract Given an identity ID, the algorithm computes $Q_{\mathrm{ID}} = H_2(\mathrm{ID})$ and $D_{\mathrm{ID}} = sH_2(\mathrm{ID})$ and outputs D_{ID} as a private key of the identity ID corresponding to $Q_{\mathrm{ID}} = H_2(\mathrm{ID})$.

Signing Given a secret key D_{ID} and a message m, pick a random number $r \in \mathbb{Z}/\ell\mathbb{Z}$ and output a signature $\sigma = (U, V)$ where $U = rP$, $h = H_1(m, U)$, and $V = rQ_{\mathrm{ID}} + hD_{\mathrm{ID}}$.

Verification Given a signature $\sigma = (U, V)$ of a message m for an identity ID, compute $h = H_1(m, U)$. The signature is accepted if and only if $e(P, V) = e(Q_{\mathrm{ID}}, U + hP_{pub})$.

The proposed scheme is secure under the assumption that the CDHP is hard as in the following theorem.

Theorem 2. *Let $\mathcal{F}_0$ be a forger which performs an existential forgery under an adaptively chosen message and* ID *attack against our* ID*-based scheme within a time bound T_0 with probability ϵ_0 in random oracle model. The forger $\mathcal{F}_0$ can ask queries to the oracles H_1, H_2,* **Extract** *and* **Signing** *at most q_{H_1}, q_{H_2}, q_E, and q_S-times, respectively. Assume that $\epsilon_0 \geq (10(q_S+1)(q_S+q_{H_1})q_{H_2})/(\ell-1)$, then the CDHP can be solved with probability $\geq$ 1/9 and within running time $\leq (23q_{H_1}q_{H_2}T_0)/\left(\epsilon_0\left(1-\frac{1}{\ell}\right)\right)$ where ℓ is a security parameter.*

Using the forking lemma [26] and [8, Lemma 1], we can prove this theorem. We discuss the rigorous proof of this theorem in the Appendix.

4.2 Security of Batch Verifications

In the our ID-based signature scheme, secure batch verification of Type 3 has possible and that of Type 2 is the same performance with the Cha-Cheon scheme. Given k signatures $(\mathrm{ID}_1, m_1, U_1, h_1, V_1), \cdots, (\mathrm{ID}_k, m_k, U_k, h_k, V_k)$, we can do batch verifications as follows:

– Compute $Q_i = H_2(\mathrm{ID}_i)$ and $h_i = H_1(m_i, U_i)$ for all $i = 1, \cdots, k$. Check whether

$$e\left(P, \sum_{i=1}^{k} V_i\right) = \prod_{i=1}^{k} e\left(Q_i, U_i + h_i P_{pub}\right).$$

Now we discuss the security of batch verification of our scheme. At first we show the security of batch verification of Type 3. To reduce the adaptively chosen ID attack to the given ID attack in the case of batch verification of Type 3, we need the following lemma:

Lemma 2. *If there is a k-batch forger $\mathcal{F}_0$ of Type 3 under an adaptively chosen message and ID attack to our scheme within time bound T_0 with probability ϵ_0, then there is a k-batch forger $\mathcal{F}$ of Type 3 under an adaptively chosen*

message and given ID attack within time bound $T \leq T_0$ *with the probability* $\epsilon \leq \epsilon_0 \left(1 - \frac{k}{\ell}\right) \left(\frac{k}{q_{H_2}+k}\right)$, *where* q_{H_2} *is the maximum number of queries to* H_2 *asked by* $\mathcal{F}_0$ *and* k *is the maximum number of signatures to be aggregated. In addition, the number of queries to hash functions,* **Extract** *and* **Signing** *asked by* $\mathcal{F}_0$ *are the same as those of* $\mathcal{F}$.

We show the proof of the Lemma 2 in the Appendix.

Now in random oracle model we show the security of batch verification of Type 3 under an adaptively chosen message and given ID attack.

Lemma 3. *Let* $\mathcal{F}$ *be* k*-batch forger of Type 3 which performs an existential forgery under an adaptively chosen message and given* ID *attack against our scheme within a time bound* T *with probability* ϵ *in random oracle model. The forger* $\mathcal{F}$ *can ask queries to the oracles* H_1, H_2, **Extract** *and* **Signing** *at most* q_{H_1}, q_{H_2}, q_E, *and* q_S*-times, respectively. If* $\epsilon \geq (10k(q_S+1)(q_S+k \cdot q_{H_1}))/\ell$, *then the CDHP can be solved with probability* $\geq 1/9$ *and within running time* $\leq (23 q_{H_2} k T)/\epsilon$.

Proof. We construct an algorithm $\mathcal{C}$ using the forger $\mathcal{F}$ to solve the CDHP. We assume that P, aP, bP are given as the CDHP instances. The algorithm $\mathcal{C}$ simulates a real signer to obtain signatures which pass the batch verification from $\mathcal{F}$. If $\mathcal{C}$ does not fail this simulation, he gets multiple signatures which pass the batch verification and using the general oracle replaying technique, it can solve the CDHP. In Setup, the algorithm $\mathcal{C}$ fixes a target identity ID_0, and put $P_{pub} = aP$.

Note that ID**-Hash Query, Extract Query, Message-Hash Query**, and **Signing Query** are the same as the single signature case. After the queries, if the simulation does not fail, the forger $\mathcal{F}$ outputs other $n-1$ ID's and n multiple signatures

$$\sigma_i = (m_i, U_i, h_i, V_i), \quad i = 1, 2, \cdots, n$$

where $n \leq k$.

Then $\mathcal{C}$ replays the oracles and obtains another $n' - 1$ ID$'$'s and n' multiple signatures $\sigma'_i = (m'_i, U'_i, h'_i, V'_i)$, $i = 1, 2, \cdots n'$, where $n' \leq k$. By the forking lemma, the replay succeeds with the probability $\geq 1/9$ and the running time $\leq (23 q_{H_2} T)/\epsilon$. Note that we may assume $h_1 \neq h'_1$ since the probability of collision of two random numbers is negligible. Since the random commitment r is fixed before the hash queries of a message, the corresponding random commitment of σ must be the same with that of σ by the forking lemma. That is, we have $\mathrm{ID}_i = \mathrm{ID}'_j = \mathrm{ID}_0$ and $U_i = U'_j$ for some $i \in \{1, \cdots, n\}$ and $j \in \{1, \cdots, n'\}$ without loss of generality let $i = j' = 1$. And according to the **Extract Query**, $\mathcal{C}$ knows each secret key D_i corresponding to ID_i except that of ID_1 and by **ID-Hash Query**, $\mathcal{C}$ knows discrete log of each $Q_i = H_2(\mathrm{ID}_i)$, x_i, except that

of $H_2(\mathrm{ID}_0)$. Hence from

$$V = \sum_{i=1}^{n} V_i = \sum_{i=1}^{n} (r_i Q_i + h_i D_i) = \sum_{i=1}^{n} \{x_i(r_i P) + h_i D_i\},$$
$$V' = \sum_{i=1}^{n'} V'_i = \sum_{i=1}^{n'} \{x'_i(r'_i P) + h'_i D'_i\},$$

compute $\overline{V} = V - V' - \sum_{i=2}^{n} V_i - \sum_{i=2}^{n'} V'_i = (h_1 - h'_1)D_1$ so $(h_1 - h'_1)^{-1}\overline{V} = D_1 = abP$ as desired. The total running time is bounded by the running time of the forking lemma. □

As the same reason with the Cha-Cheon scheme, the batch verification of Type 1 has the same performance with that of Type 3. In Type 2 signatures, the performance of our scheme is the same as that of the Cha-Cheon scheme.

From the Lemma 2, Lemma 3 and the Theorem 1, we obtain the following result.

Theorem 4 *Let $\mathcal{F}_0$ be a k-batch forger which performs an existential forgery under an adaptively chosen message and* ID *attack against our* ID*-based scheme with probability ϵ_0 within a time bound T_0 in random oracle model. The forger $\mathcal{F}_0$ can ask queries to the oracles H_1, H_2,* **Extract** *and* **Signing** *at most q_{H_1}, q_{H_2}, q_E, and q_S-times, respectively.*

- *In the Type 1 or 3 case, if $\epsilon_0 \geq (10(q_S+1)(q_S+q_{H_1})(q_{H_2}+k)q_{H_2})/k(\ell-k)$, then the CDHP can be solved with probability $\geq$ 1/9 and within running time $\leq (23q_{H_1}(q_{H_2}+k)T_0)/\left(\epsilon_0 k\left(1-\frac{k}{\ell}\right)\right)$.*
- *In the Type 2 case, if $\epsilon_0 \geq (12V_{q_{H_1},k} + 6(q_{H_1} + k \cdot q_S)^2)q_{H_2}/(\ell-1)$, then the CDHP can be solved with probability $\geq$ 1/9 and within running time $\leq 144823V_{q_{H_1},k}(1+q_S)q_{H_2}T_0/\left(\epsilon_0\left(1-\frac{1}{\ell}\right)\right)$.*

4.3 Efficiency of Batch Verifications

In this section, we compare the efficiency of verifications of k individual Cha-Cheon signatures with that of k-batch verification of Type 3 in our scheme. Here we assume that we use an elliptic curve with an admissible Tate pairing as a gap Diffie-Hellman group.

To estimate the performance of our scheme, we first present experimental results for the cost of several cryptographic primitives in Table 1. We used Miracl library v.4.8.2 [17] in P3-977 MHz with 512 Mbytes memory. In MapToPoint and Pairing, we considered a subgroup of order q in a supersingular elliptic curve E over $\mathbb{F}_p$, where p is a 512 bit prime and q is a 160 bit prime. Note that the pairing value belongs to a finite field of 1024 bits.

To verify a single Cha-Cheon signature, we need to compute two pairings, a scalar multiplication in an elliptic curve and a MapToPoint the total running time is about 73.17ms. So the running time to verify all individual signatures

Table 1. Cost of basic operations

Function	modulus (bits)	exponent (bits)	performance (msec)
Scalar Mul. in EC	512	160	7.33
MapToPoint	512	(160)	2.42
Pairing	512	(160)	31.71

signed by k distinct signers on k distinct messages is about $73k$ms. In the batch verification of Type 3 k multiple signatures using our scheme, we need to compute $k+1$ pairings, k scalar multiplications, k MapToPoints, which takes about $(41k+32)$ms. Thus if k is large, we can save about *half* of the verification time. In the batch verification of Type 2 k multiple signatures, we need to compute only two pairings, one scalar multiplication and one MapToPoint, which takes about 73ms. Thus the verification cost of the batch verification of Type 2 is almost that of a single signature.

5 Conclusion

In this paper, we classified batch verifications into three types, Type 1, 2, and 3, according to the number of signers and messages, and discussed security of previous well-known ID-based signature schemes in each type of batch verification. We have shown that the previous ID-based signature schemes are not secure in batch verifications of Type 1 and 3. We also proposed a new ID-based signature scheme admitting secure batch verification. The batch verification of Type 2 in our scheme has the same security reduction as in the previous schemes, and those of Type 1 and 3 are secure against existential forgery, under the adaptively chosen message and ID attack in random oracle model. Finally we discussed the efficiency of batch verification of Type 3 in our scheme.

References

1. P. Barreto, H. Kim, B. Lynn and M. Scott. Efficient Algorithms for Pairing-Based Cryptosystems. *Advances in Cryptology - Crypto 2002*, LNCS Vol. 2442, pp. 354–368, Springer-Verlag, 2002.
2. M. Bellare, J. Garay, and T. Rabin. Fast Batch Verification for Modular Exponentiation and Digital Signatures. *Advances in Cryptology - Eurocrypt'98*, LNCS Vol. 1403, pp. 236–250, Sringer-Verlag, 1998.
3. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and Verifiably Encrypted Signatures from Bilinear Maps. *Advances in Cryptology - Eurocrypt 2003*, LNCS Vol. 2656, pp. 416–432, Springer-Verlag, 2003.
4. D. Boneh, B. Lynn, and H. Shacham. Short signature from the Weil pairing. *Advances in Cryptology - Asiacrypt 2001*, LNCS Vol. 2248, pp. 514–531, Springer-Verlag, 2001. The extended version is available at `http://crypto.stanford.edu/~dabo/abstracts/weilsigs.html`.

5. X. Boyen. Multipurpose Identity-Based Signcryption - A Swiss Army Knife for Identity-Based Cryptography. *Advances in Cryptology - Crypto 2003*, LNCS Vol. 2729, pp. 383–399, Springer-Verlag, 2003.
6. A. Boldyreva. Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme. *Proceedings of PKC 2003*, LNCS Vol. 2567, pp. 31–46, Springer-Verlag, 2003.
7. C. Boyd and C. Pavlovski. Attacking and Repairing Batch Verification Schemes. *Advances in Cryptology - Asiacrypt 2000*, LNCS Vol. 1976, pp. 58–71, Springer-Verlag, 2000.
8. J. Cha and J. Cheon. An ID-based Signature from Gap-Diffie-Hellman Groups. *Public Key Cryptography - PKC 2003*, LNCS Vol. 2567, pp. 18–30, Springer-Verlag, 2003.
9. Y. Desmedt and J. Quisquater. Public-key Systems based on the Difficulty of Tampering. *Advances in Cryptology - Crypto'86*, LNCS Vol. 263, pp. 111–117, Springer-Verlag, 1987.
10. U. Feige, A. Fiat, and A. Shamir. Zero-knowledge Proofs of Identity. *J. Cryptology*, Vol. 1, pp. 77–94, 1988.
11. A. Fiat. Batch RSA. *J. Cryptology*, Vol. 10, No. 2, pp. 75–88, Springer-Verlag, 1997. A preliminary version appeared in *Advances in Cryptology - Crypto'89*, LNCS Vol. 435, pp. 175–185, Springer-Verlag, 1989.
12. F. Zhang and K. Kim, Efficient ID-based blind signature and proxy signature from bilinear pairings, *ACISP 03*, LNCS Vol. 2727, pp. 312–323, Springer-Verlag, 2003.
13. A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. *Advances in Cryptology - Crypto '86*, LNCS Vol. 263, pp. 186–194, Springer-Verlag, 1987.
14. F. Hess. Efficient Identity Based Signature Schemes Based on Pairings. *Selected Areas in Cryptography - SAC 2002*, LNCS Vol. 2595, pp. 310–324, Springer-Verlag, 2002.
15. J. Herranz and G. Sáez. Forking Lemmas in Ring Signatures' Scenario. *Progress in Cryptology - INDOCRYPT 2003*, LNCS Vol. 2904, pp. 266–279, Springer-Verlag Heidelberg, 2003.
16. K. Itakura and K. Nakamura. A Public-key Cryptosystem Suitable for Digital Multisignatures. *NEC Research and Development*, Vol. 71, pp. 1–8, 1983.
17. Shamus Software Ltd. Miracl: Multiprecision integer and rational arithmetic c/c++ library. `http://indigo.ie/~mscott/`.
18. D. M'Raithi and D. Naccache. Batch Exponentiation - A Fast DLP based Signature Generation Strategy. *ACM Conference on Computer and Communications Security*, pp. 58–61, ACM, 1996.
19. E. Mykletun, M. Narasimha and G. Tsudik. Providing Efficient Data Integrity Mechanisms in Outsourced Databases. *Network and Distributed System Security (NDSS)*, 2004.
20. S. Micali, K. Ohta and L. Reyzin, Accountable-subgroup Multisignatures. *On proceedings of CCS 2001*, pp. 245–254, ACM, 2001.
21. U. Maurer and Y. Yacobi. Non-interactive Public-key Cryptography. *Advances in Cryptology - Eurocrypt'91*, LNCS Vol. 547, pp. 458–460, Springer-Verlag, 1992.
22. D. Naccache, D. M'Raithi, S. Vaudenay, and D. Raphaeli. Can D.S.A be Improved? Complexity trade-offs with the Digital Signature Standard. *Advances in Cryptology - Eurocrypt'94*, LNCS Vol. 950, pp. 77–85, Springer-Verlag, 1994.
23. K. Ohta and T. Okamoto. A Digital Multisignature Scheme based on the Fiat-Shamir Scheme. *Advances in Cryptology - ASIACRYPT'91*, LNCS Vol 739. pp. 75–79, Spring-Verlag, 1991.

24. K. Ohta and T. Okamoto. Multi-signature Schemes Secure against Active Insider Attacks. *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, Vol. E-82-A, No. 1, pp. 21–31, 1999.
25. K. Paterson. ID-based Signatures from Pairings on Elliptic Curves. *Electronics Letters*, Vol. 38, No. 18, pp. 1025–1026, 2002.
26. D. Pointcheval and J. Stern. Security Arguments for Digital Signatures and Blind Signatures. *J. Cryptology*, Vol. 13, No. 3, pp. 361–396, 2000. A preliminary version has appeared in *Advances in Cryptology - Eurocrypt'96*, LNCS Vol. 1070, pp. 387–398, Springer-Verlag, 1996.
27. A. Shamir. Identity-base Cryptosystems and Signature Schemes. *Advances in Cryptology - Crypto'84*, LNCS Vol. 196, pp. 47–53, Springer-Verlag, 1985.
28. C. Schnorr. Efficient Identification and Signatures for Smart Cards. *Advances in Cryptology - Crypto'89*, LNCS Vol. 435, pp. 239–252, Springer-Verlag, 1989.
29. H. Tanaka. A Realization Scheme for the Identity-based Cryptosystem. *Advances in Cryptology - Crypto'87*, LNCS Vol. 293, pp. 340–349, Springer-Verlag, 1987.
30. S. Tsuji and T. Itoh. An ID-based Cryptosystem based on the Discrete Logarithm Problem. *IEEE Journal of Selected Areas in Communications*, Vol. 7, pp. 467–473, 1989.
31. F. Zhang and K. Kim. Efficient ID-based Blind Signature and Proxy Signature from Bilinear Pairings. *ACISP 03*, LNCS Vol. 2727, pp. 312–323, Springer-Verlag, 2003.

Appendix: Security Proof

Proof of Theorem 2.

Using [8, Lemma 1], we can reduce the forger $\mathcal{F}_0$ to $\mathcal{F}$ an adaptively chosen message and given ID attack within time bound $T \leq T_0$ with the probability $\epsilon \leq \epsilon_0(1-\frac{1}{\ell})/q_{H_2}$. We construct an algorithm $\mathcal{C}$ using $\mathcal{F}$ to solve the CDHP. We assume that P, aP, and bP are given. Since the forger $\mathcal{F}$ is an adaptively chosen message attacker, he can access to the hash oracles, the extraction oracle, and the signing oracle, and ask at most q_{H_1}, q_{H_2}, q_E, and q_S queries for each oracles respectively. The algorithm $\mathcal{C}$ simulates a real signer to get a valid signature from the forger $\mathcal{F}$. If $\mathcal{C}$ does not fail this simulation, he gets a valid signature, and using the oracle replaying technique he can solve the CDHP.

We may assume the forger is well-behaved in the following sense: A forger $\mathcal{F}$ makes a **Extract** query for an ID only if an H_2 query has been made before for the ID. Also **Signing** query is made for a message m only if a H_1 queries has been made before for the m.

Then the algorithm $\mathcal{C}$ puts $P_{pub} = aP$ and performs the following game with the forger $\mathcal{F}$ for a fixed identity ID as follows:

ID-Hash Query When $\mathcal{F}$ makes an ID-hash query ID_i, $\mathcal{C}$ gives to $\mathcal{F}$ an answer $H_2(\mathrm{ID}_i) = bP$ if $\mathrm{ID}_i = \mathrm{ID}$ and $H_2(\mathrm{ID}_i) = x_iP$ for $x_i \in_R \mathbb{Z}/\ell$ otherwise.

Extract Query When $\mathcal{F}$ makes an extract query for ID_{i_k}, $\mathcal{C}$ gives $x_{i_k}P_{pub} = x_{i_k}(aP)$ as the secret key corresponding to $H_2(\mathrm{ID}_{i_k})$ for an identity ID_{i_k}. Note that $\mathcal{F}$ must not ask the secret key corresponding to the $bP = H_2(\mathrm{ID})$.

Message-Hash Query $\mathcal{F}$ makes q_H message-hash queries. For the j-th hash query $\mathcal{Q}_j$, $\mathcal{C}$ chooses a random value $h_j \in \mathbb{Z}/\ell$ and gives to $\mathcal{F}$ as the hash value of $\mathcal{Q}_j$ for $j = 1, \cdots q_{H_1}$ and stores them as $H_1(\mathcal{Q}_j) = h_j$.

Signing Query If $\mathcal{F}$ asks the signature on m_{j_t} of ID_{i_t}, $\mathcal{C}$ chooses a random value $r_t \in \mathbb{Z}/\ell$ responses

$$Sign(\mathrm{ID}_{i_t}, m_{j_t}) = (\mathrm{ID}_{i_t}, m_{j_t}, U_t, h_t, V_t),$$

where $U_t = r_t P - h_t P_{pub}$ and $V_t = r_t(x_{i_t} P)$ for $t = 1, \cdots, q_S$. Since $(P, H_2(\mathrm{ID}_{i_t}), U_t + h_t P_{pub}, V_t)$ is a valid Diffie-Hellman tuple, these signatures pass the verification algorithm.

If the simulation does not fail, the forger $\mathcal{F}$ outputs a valid signature $(\mathrm{ID}, m, U, h, V)$ with probability ϵ. After a replay of the forger $\mathcal{F}$, apply the forking lemma in [26]. Then $\mathcal{C}$ obtains two valid signatures $\sigma = (\mathrm{ID}, m, U, h, V)$ and $\sigma' = (\mathrm{ID}, m, U, h', V')$ such that $h \neq h'$ with probability $\geq 1/9$ within the time $23 q_{H_1} T/\epsilon$. $\mathcal{C}$ can easily obtain the value abP from

$$\frac{(hD_{\mathrm{ID}} - h' D_{\mathrm{ID}})}{h - h'} = D_{\mathrm{ID}} = abP.$$

By the forking lemma [26] and [8, Lemma 1], we obtain the result of this theorem. □

Proof of Lemma 2.

Proof. We assume, without loss of generality, the forger $\mathcal{F}_0$ has an extract queries for any ID at most once. We consider an algorithm $\mathcal{F}$ that performs the following simulation:

Setup $\mathcal{F}$ chooses a random number $r \in \{1, \cdots, q_{H_1}\}$. Let ID_i be the $\mathcal{F}_0$'s i-th H_2-query and $\mathrm{ID}'_i = ID$ if $i = r$ and $\mathrm{ID}'_i = \mathrm{ID}_i$ otherwise. Let $H'_2(ID_i) = H_2(ID'_i)$, **Extract**$'(ID_i)$=**Extract**(ID'_i) and **Signing**$'(ID_i, m_i)$=**Signing**(ID'_i, m_i)

Queries If $\mathcal{F}_0$ makes the H_1, H_2 hash queries and **Extract**, **Signing** queries, then $\mathcal{F}$ computes H_1, H'_2, **Extract**$'$and **Signing**$'$ as above and answers the results.

If the simulation does not fail, $\mathcal{F}_0$ outputs signatures $\sigma_i = (\mathrm{ID}^i_{out}, m_i, U_i, h_i, V_i)$ where $i = 1, \cdots k$ with probability ϵ_0. Finally, if $\mathrm{ID}^i_{out} = ID$ for some $i = 1, \cdots, k$ and $\mathrm{ID}^j_{out} \neq \mathrm{ID}$ for all $j \neq i$ and all σ_is are valid signatures, then $\mathcal{F}$ outputs σ_i where $i = 1, \cdots, k$. Otherwise the simulation fails.

Since the output distributions of H'_2, **Extract**$'$, **Signing**$'$-queries are not distinguishable those of original ones, we know

$$\Pr[\text{For all } i \in \{1, \cdots, k\},\ \sigma'_i\text{s are valid}] \geq \epsilon.$$

Since we consider the hash functions as the random oracles, we obtain the following result.

$$\Pr[\mathrm{ID}^j_{out} = ID_i \text{ for some } j = 1, \cdots, k \text{ and } i = 1, \cdots q_{H_2} \\ \mid \text{For all } i \in \{1, \cdots, k\},\ \sigma'_i\text{s are valid}] \geq \left(1 - \frac{1}{\ell}\right)^k \geq 1 - \frac{k}{\ell}$$

Furthermore since the randomness of r, we have the following inequality.

$$\Pr[\mathrm{ID}^i_{out} = ID_r \text{ for some } i = 1, \cdots, k \text{ and } \mathrm{ID}^j_{out} \neq \mathrm{ID} \text{ for some } j = 1, \cdots, i{-}1, \\ i+1, \cdots, k \mid \mathrm{ID}^j_{out} = ID_i \text{ for some } j = 1, \cdots, k \text{ and } i = 1, \cdots q_{H_2}] \\ \geq \frac{q_{H_2}-1\mathrm{H_{k-1}}}{\mathrm{q_{H_2}H_k}} \geq \frac{k(q_{H_2}-1)}{(q_{h_2}+k-1)(q_{H_2}+k-2)} \geq \frac{k}{2(q_{H_2}+k)}$$

Finally, summarizing these, we get the following result as desired:

$$\Pr[\mathrm{ID}^i_{out} = ID_r = ID \text{ for some } i = 1, \cdots, k \text{ and } \mathrm{ID}^j_{out} \neq \mathrm{ID} \text{ for some} \\ j = 1, \cdots, i-1, i+1, \cdots, k \text{ and For all } i \in \{1, \cdots, k\}, \\ \sigma'_i\text{s are valid}] \geq \epsilon \cdot \left(1 - \frac{1}{\ell}\right) \cdot \frac{k}{2(q_{H_2}+k)}.\square$$