More Efficient Two-Round Multi-Signature Scheme with Provably Secure Parameters

Kaoru Takemure[1,2], Yusuke Sakai[2], Bagus Santoso[1], Goichiro Hanaoka[2], and Kazuo Ohta[1,2]

[1] The University of Electro-Communications, Japan
[2] National Institute of Advanced Industrial Science and Technology (AIST), Japan

Abstract. In this paper, we propose the first two-round multi-signature scheme that can guarantee 128-bit security under a standardized EC in concrete security without using the Algebraic Group Model (AGM). To construct our scheme, we introduce a new technique to tailor a certain special homomorphic commitment scheme for the use with the Katz-Wang DDH-based signature scheme. We prove that an EC with at least a 321-bit order is sufficient for our scheme to have the standard 128-bit security. This means that it is easy for our scheme to implement in practice because we can use the NIST-standardized EC P-384 for 128-bit security. The signature size of our proposed scheme under P-384 is 1152 bits, which is the smallest size among the existing schemes without using the AGM. Our experiment on an ordinary machine shows that for signing and verification, each can be completed in about 65 ms under 100 signers. This shows that our scheme has sufficiently reasonable running time in practice.

Keywords: Multi-signatures · Concrete security · Tight security.

1 Introduction

In a *multi-signature* scheme [23], for a single common message m, multiple parties cooperatively generate a signature, known as a multi-signature, which is basically a combination of multiple individual signatures on m where each is created by each party using its own signing key. An essential property of the multi-signature is that its size is kept constant independently of the number of parties. Multi-signature schemes based on several hardness problems are proposed so far, e.g., the DL (discrete logarithm)-based schemes [2, 6, 8, 16, 29, 33, 34, 37, 38], pairing-based schemes [10, 11, 28, 31, 41], and lattice-based schemes [5, 12, 15, 19, 32].

In this research, we focus on DL-based multi-signature schemes which can be implemented under the elliptic curves used to implement standard digital signature schemes, e.g., the ECDSA [36] and the Schnorr signature scheme [43], used in cryptocurrencies, e.g., Bitcoin.

For multi-signatures, the primary desirable features are the followings. The first one is the security in the plain public-key (PPK) model, which allows an adversary to make cosigners' public keys maliciously without knowing the signing keys. The second is key aggregation property, which allows aggregating a set of public keys into a single short key when signatures are verified. The third is a small number of rounds of communication for signing. Moreover, a scheme achieving a small signature size (independent from the number of signers) while achieving these three properties is desirable.

For DL-based multi-signature schemes, Bellare and Neven proposed the first *three-round* scheme, which is the scheme with *three* rounds of communication for signing, proven secure in the PPK model [8]. Moreover, Maxwell et al. proposed the first three-round scheme with key aggregation [33]. In recent years, several *two-round* schemes, achieving security in the PPK model and the support key aggregation, are proposed [2, 6, 16, 29, 37, 38].

1.1 Importance of Concrete Security for Parameter Choice

In theoretically, a security proof of a cryptosystem consists of a reduction from solving some computational problem to breaking the cryptosystem under a defined adversarial model. From the security proof, usually, we can derive a relation between $T_A = t_A/\varepsilon_A$ and $T_P = t_P/\varepsilon_P$, where t_A and ε_A are the adversary's running time and success probability for breaking the cryptosystem and t_P and ε_P are the algorithm's running time and success probability for solving the computational problem. A typical relation between T_P and T_A is as follows: $T_A \geq T_P/\Phi$, where Φ is often referred to as the *reduction loss*.

When we derive the size of parameters, e.g., an order of the underlying group in a DL-based scheme, for guaranteeing the security of the cryptosystem in practice, Φ was often disregarded. However, this disregard sometimes makes schemes vulnerable. An example of this vulnerability is shown by the recent work of Kales and Zaverucha [24] which demonstrated an attack on the MQDSS signature scheme [42]. Their attack exploits the fact that the parameter of MQDSS was derived without considering Φ. Therefore, it is important to derive the size of parameters by considering Φ based on the security proof, and thus we should implement cryptosystems with *provable secure parameters*.

A small Φ is preferable in practice because it does not let the problem mentioned above occur in the first place. Also, if Φ is large, we need to ensure that T_P is sufficiently large so that the derived lower bound of T_A, i.e., T_P/Φ, is not too small to have a practical meaning. Usually, the only way to make T_P larger is by setting larger parameters, which means higher costs for implementation in practice. If Φ is a relatively small constant value independent of the parameters of the cryptosystem and the adversary, we say that the security proof is *tight*.

Also, it is difficult for a scheme with a large Φ to use the standardized cryptographic tools. Specifically, for the DL-based schemes, a elliptic curve (EC) with a 256-bit prime order is required to guarantee 128-bit security, but it is only applied to tightly secure schemes. The scheme with a large Φ requires an EC with an order larger than 256-bit. We have standardized ECs with such order, e.g., NIST P-384 and P-521. However, for the scheme with very large Φ, such ECs are not sufficient for 128-bit security, and then, we need to design a new desireable EC. This makes the implementation difficult and less reliable.

Table 1. Detailed Performance Comparison among Two-Round Multi-Signature Schemes.[4]

Scheme	Assumption	Model	Rewinding	Sig. Size	Com. Comp.	$\lvert q\rvert_{128}$ (bit) Curve	$\lvert\widetilde{\sigma}\rvert_{128}$ (bit) $\lvert\widetilde{\sigma}\rvert_{\mathrm{EC}}$ (bit)	$\lvert\mathrm{CC}\rvert_{128}$ (bit) $\lvert\mathrm{CC}\rvert_{\mathrm{EC}}$ (bit)	Key Agg.
MuSig2 ($\nu = 2$) [37]	OMDL	AGM+ROM	Not Needed	$\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_q\rvert$	$2\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_q\rvert$	257 P-256	515 513	773 770	Yes
DWMS [2]	OMDL			$\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_q\rvert$	$2\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_q\rvert$	257 P-256	515 513	773 770	
HBMS-AGM [6]	DL			$\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_q\rvert$	$\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_q\rvert$	257 P-256	772 769	772 769	
LK [29]	DL	AGM+NPROM		$3\lvert\mathbb{Z}_q\rvert$	$\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_q\rvert$	258 P-256	774 768	775 769	
MuSig-DN [38]	DL, DDH, PRNG zk-SNARKs, PRF	ROM	Used	$\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_q\rvert$	$2\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_q\rvert + \lvert\pi\rvert$	740 Not Exist	1481 -	- -	
MuSig2 ($\nu \geq 4$) [37]	OMDL			$\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_q\rvert$	$\nu\lvert\mathbb{G}\rvert + \lvert\mathbb{Z}_q\rvert$	750 Not Exist	1501 -	3754 -	
HBMS [6]	DL			$\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_q\rvert$	$\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_q\rvert$	986 Not Exist	2959 -	2959 -	
mBCJ [16]	DL			$\lvert\mathbb{G}\rvert + 3\lvert\mathbb{Z}_q\rvert$	$2\lvert\mathbb{G}\rvert + 3\lvert\mathbb{Z}_q\rvert$	544 Not Exist	2177 -	2722 -	No
Ours	DDH	ROM	Not Needed	$3\lvert\mathbb{Z}_q\rvert$	$2\lvert\mathbb{G}\rvert + 2\lvert\mathbb{Z}_q\rvert$	321 P-384	963 1152	1286 1538	Yes

* Column 2 shows the security assumptions. Column 3 shows whether idealized models are used for a cyclic group and hash functions. Column 4 shows whether the reduction in the security proof uses the rewinding technique. Columns 5 and 6 show the size of the multi-signature and elements sent in the signing protocol per a signer, respectively. Column 7 shows the required underlying group size $|q|_{128}$ and the NIST standardized EC that enables a parameter choice with 128-bit security, which is called the recommended EC hereafter. Column 8 shows the signature sizes $|\widetilde{\sigma}|_{128}$ and $|\widetilde{\sigma}|_{\mathrm{EC}}$ under the $|q|_{128}$-bit EC group and the recommended EC, respectively. Column 9 shows the communication complexities $|\mathrm{CC}|_{128}$ and $|\mathrm{CC}|_{\mathrm{EC}}$ under the $|q|_{128}$-bit EC group and the recommended EC. Column 10 shows whether each scheme allows key aggregation. $\mathbb{G}$ and $\mathbb{Z}_q$ indicate the underlying group $\mathbb{G}$ of a prime order q and the ring of integers modulo q, respectively. We assume that the sizes of $|\mathbb{G}|$ and $|\mathbb{Z}_q|$ over a q-bit EC are $q+1$ and q bits, respectively. ROM and NPROM indicate the random oracle model and the non-programmable random oracle model. $|\pi|$ is the size of the zk-SNARK proof. For MuSig-DN, we write "-" in Column 9 because the size of $|\pi|$ considering concrete security is explicitly unknown.

For schemes with a small Φ, it is easy to use the standardized tools because we do not have to care about the above-mentioned problems. Specifically, for the DL-based schemes, we can use the standardized ECs, e.g., NIST P-256 and Secp256k1, for 128-bit security. Also if Φ is small, even though Φ is not constant, we may be able to avoid the inconvenient situation where there is no suitable standardized tool.

1.2 Concrete Security of Existing Multi-Signature Schemes

Tightness of Multi-Signature Schemes. Here, we review the existing two-round multi-signature schemes in terms of the tightness of a reduction.

The existing two-round multi-signature schemes can be categorized into two types: The first type is the schemes with a non-tight reduction (namely, having a large reduction loss) and the second type is the schemes with a tight reduction. The schemes of the first type include MuSig-DN [38], MuSig2 ($\nu \geq 4$) [37], HBMS [6], and mBCJ [16], while the schemes of the second type include MuSig2 ($\nu = 2$) [37], DWMS [2], HBMS-AGM [6], and LK [29].

[4] For MuSig2, ν is a unique parameter. In this comparison, mBCJ is a variant that is secure in the plain public-key model, not the original one proposed in [16].

When taking tightness into consideration, even for 128-bit security, the first-type schemes require an elliptic curve (EC) with an order *larger than 256-bit.* This is because a non-tight proof leaves a gap between the hardness of the discrete logarithm (DL) problem (or some hard problem) in the underlying group and the (proven) hardness of breaking the signature scheme in question. Thus, there may be an attack against the scheme which falls into this gap. To make such an attack inefficient provably, one needs to set the order of the group to be larger than 256 bits, even for 128-bit security. Table 1 shows how large the order of the group should be in order to provably ensure 128-bit security for the above-mentioned schemes.

As Table 1 shows, each scheme of the first type requires a very large order of a group. MuSig-DN, MuSig2 ($\nu \geq 4$), HBMS, and mBCJ, respectively, require 740-bit, 750-bit, 986-bit, and 544-bit groups. Importantly, these schemes no longer have standardized curves that provably ensure 128-bit security.

The cause of the large reduction losses of these schemes is that to prove the security based on the DL assumption, the reduction performs the *rewinding* of the adversary, like the security proof of the Schnorr signature scheme. Moreover, in schemes with key aggregation, the number of rewindings has to be increased. Thus, these schemes have larger reduction losses than that of the Schnorr signature scheme.

The schemes of the second type achieve tight security by using the Algebraic Group Model (AGM) [17], which is a very idealized model of computation. The schemes allow us to use an EC of a small order, e.g., 256-bit. However, the reliance on the AGM needs more care because recent research [25,44] shows that the reliability of the AGM is still not well-understood. Indeed, Zhandry showed the one-time message authentication code that is secure in the AGM but insecure in the standard model [44].

The random oracle model is also an idealized model of hash functions, but the situation is rather different from the AGM. Much cryptanalytic literature investigates the possibility of distinguishing a concrete hash function from a random oracle by finding a dedicated input-output correlation beyond (target) collision resistance or preimage resistance [3,21,26,27]. These lines of research provide a more fine-grained understanding of how far (or near) concrete hash functions are from a random oracle.

1.3 Our Contribution

In this paper, we propose a two-round multi-signature scheme from the decisional Diffie-Hellman (DDH) assumption and the random oracle model, without using the AGM. This scheme guarantees 128-bit security under a standardized EC, e.g., NIST P-384. Thus, Our scheme is the first scheme that can guarantee 128-bit security under a standardized EC without using AGM. Moreover, our scheme is proven secure in the PPK model and supports key aggregation.

To achieve a scheme with the following both properties: two rounds of communication for signing and a small reduction loss, we construct our scheme based on the Katz-Wang signature scheme [22] by applying the technique of HBMS.

HBMS uses a certain special homomorphic commitment scheme to achieve a two-round signing protocol. However, this commitment scheme is specialized to use together with the Schnorr signature scheme. Therefore, we cannot apply this commitment scheme to our case. To overcome this, we introduce a new technique to tailor a certain special homomorphic commitment scheme for the use with the Katz-Wang DDH-based signature scheme.

We describe the features of our proposed scheme by comparing with the existing schemes without using the AGM in below.

Although ours does not achieve tight security, i.e., a constant reduction loss, ours can reduce the reduction loss compared to the other schemes. Specifically, we prove that our scheme only needs an EC with at least 321-bit order to ensure 128-bit security. Therefore, the curve P-384 is sufficient. Using an EC with a smaller order makes several computations faster and hardware implementations easier. Existing schemes in Table 1 cannot use a standardized EC for 128-bit security. As previously described, such a situation makes implementation difficult and reduces the reliability of the parameter choice. Thus, we can conclude that the advantage of our scheme removes several obstacles to implementation.

The signature size and the communication complexity of ours are the shortest among the existing schemes. Under P-384, the size of the multi-signature is 1152 bits. Compared to MuSig-DN, MuSig2 ($\nu \geq 4$), HBMS, and mBCJ, this size is reduced by more than 22%, 23%, 60%, and 47%, respectively. Moreover, the communication complexity of ours is the shortest among them. The communication completely is 1538 bits under P-384. Compared to MuSig2 ($\nu \geq 4$), HBMS, and mBCJ, this size is reduced by more than 59%, 48%, and 43%.

We implement our scheme on an ordinary machine and measure the running time of our implementation.[5] We set the number of signers $N = 3$, 5, 10, and 15, as typical numbers of signers in a real-world Multi-Sig Wallet, and $N = 50$ and 100 as large-scale settings. For more details of the setting and the environment, see Section 5. Both the running time of the signing protocol and that of the verification under $N = 15$ are less than 10 ms. For large-scale settings, both the running time of the signing protocol and that of the verification are about 30 ms under $N = 50$, and those are about 65 ms under $N = 100$. Moreover, since our proposed scheme also supports key aggregation, by precomputing the aggregated key, both the running time of signing and that of verification can be shortened to less than 2 ms irrelevantly to N. Thus, we can consider that our scheme has a realistic running time in practice.

1.4 Related Works

Bellare and Neven proposed the first Schnorr-based three-round multi-signature scheme [8] which is secure in the PPK model. In the document [7], they also proposed a DDH-based scheme which is built from the Katz-Wang signature scheme. Maxwell et al. proposed a variant of the Bellare-Neven scheme that

[5] We do not take into account the communication time between parties because it varies largely depending on the network environment.

supports key aggregation [33]. Fukumitsu and Hasegawa proposed a DDH-based scheme with key aggregation [18].

Drijvers et al. proposed a secure Schnorr-based two-round multi-signature scheme mBCJ [16]. They constructed this scheme by applying a patch to the insecure two-round scheme BCJ [4] which uses the homomorphic (special) equivocal commitment scheme. Bellare and Dai proposed the improvement of mBCJ as HBMS [6]. Recently, Lee and Kim proposed a two-round scheme LK [29] based on HBMS and the Okamoto identification scheme [39]. The security of these schemes is proven under the DL assumption. The approach to constructing our proposed scheme based on the DDH assumption is a similar direction to mBCJ's approach. MuSig-DN [38] is the two-round multi-signature scheme with the different approach from the above schemes. Specifically, this scheme achieves a two-round signing protocol by using the pseudorandom functions (PRF), the pseudorandom number generators (PRNG), and the succinct non-interactive arguments of knowledge (SNARKs) [13] to make the signing protocol deterministic. MuSig2 [37] and DWMS [2] are also two-round schemes with different approaches from the above-mentioned schemes. They are proven secure under the one-more DL assumption.

2 Preliminaries

2.1 Notation

Unless noted otherwise, any algorithm is probabilistic. For an algorithm $\mathcal{A}$, we write $b \leftarrow\$ \mathcal{A}(\alpha_1, \ldots)$ to mean that $\mathcal{A}$ on inputs $\alpha_1, \ldots$ and a uniformly chosen random tape outputs b. For algorithms $\mathcal{A}_1, \ldots, \mathcal{A}_n$, we write $b \leftarrow\$ \langle\{\mathcal{A}_i(\alpha_{i1}, \ldots)\}_{i=1}^{n}\rangle$ to mean that each algorithm $\mathcal{A}_i$ on inputs $\alpha_{i1}, \ldots$ and a uniformly chosen random tape executes a protocol with the others, and eventually all algorithms obtain b. For a list L, we write the i-th element in L as $L[i]$. For any value a, we write $a \leftarrow b$ means the assignment of a into b.

2.2 Hardness Assumption

For a prime integer q, we denote the ring of integers modulo q by $\mathbb{Z}_q$. Let $\mathbb{G}$ be an additive cyclic group of order q and let G be a generator of $\mathbb{G}$. We denote the identity element of $\mathbb{G}$ by O.

In this paper, we use the following notations. For A, B, G, $H \in \mathbb{G}$ and $x \in \mathbb{Z}_q$, we write $(A, B)^T \leftarrow x(G, H)^T$ to mean that A and B are computed by xG and xH, respectively. Also, for A, B, G, H, Y, $Z \in \mathbb{G}$, we write $(A, B)^T \leftarrow (G, H)^T + (Y, Z)^T$ to mean that A and B are computed by $G + Y$ and $H + Z$, respectively.

Below, we recall the definition of the decisional Diffie-Hellman (DDH) assumption.

Definition 1. *The advantage* $\mathrm{Adv}_{\mathbb{G}}^{\mathrm{ddh}}(\mathcal{A})$ *of an algorithm* $\mathcal{A}$ *is defined as*

$$\begin{aligned}\mathrm{Adv}_{\mathbb{G}}^{\mathrm{ddh}}(\mathcal{A}) = |&\Pr[\mathcal{A}(G, xG, yG, xyG) = 1 : x, y \leftarrow\!\!\$\; \mathbb{Z}_q] \\ &- \Pr[\mathcal{A}(G, xG, yG, zG) = 1 : x, y \leftarrow\!\!\$\; \mathbb{Z}_q, z \leftarrow\!\!\$\; \mathbb{Z}_q \backslash \{xy\}]| \,.\end{aligned}$$

We say that an algorithm $\mathcal{A}$ (t, ε)*-solves the DDH problem in* $\mathbb{G}$ *if* $\mathcal{A}$ *runs in time at most* t *and satisfies* $\mathrm{Adv}_{\mathbb{G}}^{\mathrm{ddh}}(\mathcal{A}) \geq \varepsilon$*. We also say that* $\mathbb{G}$ *is a* (t, ε)*-DDH group if there is no algorithm that* (t, ε)*-solves the DDH problem in* $\mathbb{G}$*.*

2.3 Multi-Signatures

In this section, we show the definition and security model of the multi-signature scheme.

Definition 2. *A multi-signature scheme consists of the following three algorithms and an interactive protocol. Let* n *be the number of signers.*

$\mathbf{Pg}() \rightarrow pp$**.** *On no input, the public parameter generation algorithm outputs a public parameter* pp*.*

$\mathbf{Kg}(pp) \rightarrow (pk, sk)$**.** *On an input* pp*, the key generation algorithm outputs a public key* pk *and a secret key* sk*.*

$\langle \{\mathcal{S}(pp, i, sk_i, L, m)\}_{i=1}^{n} \rangle \rightarrow \widetilde{\sigma}$**.** *The signing protocol is executed by multiple signers who intend to sign on the common message* m*. On inputs* pp*, an index of signers* i*,* sk_i*, a public-key list* $L = \{pk_1, \ldots, pk_n\}$*, and* m*, each signer executes the signing protocol with cosigners. Finally, it outputs a multi-signature* $\widetilde{\sigma}$ *on* m*.*

$\mathbf{Vf}(pp, L, \widetilde{\sigma}, m) \rightarrow \{0, 1\}$**.** *On inputs* pp*,* L*,* $\widetilde{\sigma}$*, and* m*, the verification algorithm deterministically outputs 1 (Accept) or 0 (Reject).*

A multi-signature scheme must satisfy the following completeness property: For any message m and any n,

$$\Pr\left[\mathbf{Vf}(pp, L, \widetilde{\sigma}, m) = 1 : \begin{array}{l} pp \leftarrow\!\!\$\; \mathbf{Pg}(), \\ {}^{\forall} i \in \{1, n\}, (pk_i, sk_i) \leftarrow\!\!\$\; \mathbf{Kg}(pp), \\ L \leftarrow \{pk_i\}_{i=1}^{n}, \\ \widetilde{\sigma} \leftarrow\!\!\$\; \langle \{\mathcal{S}(pp, i, sk_i, L, m)\}_{i=1}^{n} \rangle \end{array}\right] = 1.$$

Security Definition of Multi-Signatures. Here, we show the security definition of multi-signatures. Our security model is very similar to Bellare-Neven's model [8].

Our security model is formally defined by the following three-phase game.

Setup. The challenger generates a public parameter pp by $\mathbf{Pg}()$ and a key pair (pk, sk) by $\mathbf{Kg}(pp)$. It initializes tables $T_M[\cdot]$ and $T_\Sigma[\cdot]$ to $\emptyset$. It sends pp and pk to a forger $\mathcal{F}$. $\mathcal{F}$ is allowed to access random oracles and a signing oracle. $\mathcal{F}$ can corrupt all signers who have different public keys from pk.

Signing Oracle. The challenger receives a message m, a session number I_s, and a public-key list L as a signing query from $\mathcal{F}$. Let $n_h = \{i | L[i] = pk\}$. The challenger responses as follows.

Case $|n_h| = 0$**.** The challenger returns $\perp$ to $\mathcal{F}$.

Case $|n_h| \geq 1$**.** The challenger executes the signing protocol by behaving as any honest signer corresponding to indices in n_h. Note that it behaves as each honest signer by maintaining its state and using its random tape. Let st_ℓ be the state information, kept during the protocol, of the honest signer corresponding to $\ell \in n_h$. At the end of each round of the protocol, the challenger stores $T_\Sigma[I_s] \leftarrow \{st_\ell\}_{\ell \in n_h}$.

$\mathcal{F}$ is allowed to make multiple signing queries concurrently. Note that, under the same session number, $\mathcal{F}$ is not allowed to make multiple signing queries in the same round of the signing protocol. If the signing protocol for queried m is completed, the challenger assigns $T_M[m] \leftarrow 1$.

Check. Finally, $\mathcal{F}$ outputs a public-key list L^*, a message m^*, and a forgery $\widetilde{\sigma}^*$. $\mathcal{F}$ is said to win the game if $\mathcal{F}$'s output satisfies that $pk \in L^*$, $T_M[m^*] \neq 1$, and $\mathbf{Vf}(pp, L^*, \widetilde{\sigma}^*, m^*) = 1$.

Definition 3. *For a multi-signature scheme* $\mathbf{MS}$*, let* $\mathrm{Adv}_{\mathbf{MS}}(\mathcal{F})$ *be the probability that* $\mathcal{F}$ *wins the above game. We say that* $\mathcal{F}$ $(t, q_S, q_H, N, \varepsilon)$*-breaks multi-signature scheme* $\mathbf{MS}$ *if* $\mathcal{F}$ *runs in at most* t *time, it makes at most* q_S *signing queries and* q_H *random oracle queries, the numbers of public keys included in* L *for any signing queries and a forgery are at most* N*, and* $\mathrm{Adv}_{\mathbf{MS}}(\mathcal{F}) \geq \varepsilon$*. We say a multi-signature scheme* $\mathbf{MS}$ *is* $(t, q_S, q_H, N, \varepsilon)$*-secure if there is no* $\mathcal{F}$ *that* $(t, q_S, q_H, N, \varepsilon)$*-breaks* $\mathbf{MS}$*.*

3 Proposed Scheme

We show the construction of our proposed two-round multi-signature scheme and its security. We construct our scheme by combining the Katz-Wang DDH-based signature scheme [22] to avoid rewinding and the technique of HBMS [6], a Schnorr-based two-round multi-signature scheme.

3.1 Overview of Our Proposed Scheme

Below, we describe the intuition of the construction.

Towards constructing a tightly secure two-round multi-signature scheme, we begin with the Katz-Wang (standard) signature scheme and a three-round multi-signature version of this by Bellare and Neven [7].

Recall that the Katz-Wang signature scheme employs a lossy identification protocol [1] based on the DDH assumption. The public key of this signature scheme is a Diffie-Hellman (DH) tuple. The unforgeability is proven under the DDH assumption as follows: First, we prove that when the public key is replaced with a non-DH tuple, which is called the *lossy key*, forgery is *statistically* hard using the (simulation) soundness of the Fiat-Shamir transformation of the lossy identification protocol. Namely, under the lossy key, even for a computationally unbounded adversary, the success probability of an adversary is negligible. Next, assume that there is an adversary that can perform forgery with a non-negligible probability under the real public key, i.e., a DH tuple. Notice that this assumption induces a non-negligible gap between the adversary's success probability under the lossy key and that under the real public key. Based on this, we can construct an algorithm that can solve the DDH problem by internally running (without rewinding) the adversary given an instance of the problem as the public key.

The Bellare-Neven three-round multi-signature scheme based on the DDH assumption [7] uses a random oracle as a commitment scheme. Due to this use of a random oracle, the signing protocol requires three rounds. Specifically, in the first round, each signer generates a commitment value t of a commitment R of the Schnorr protocol by using a random oracle and shares t with all cosigners. In the second round, they open their commitment R. Finally, in the third round, each signer sends his response s of the Schnorr protocol to each other and then generates a multi-signature. To generate s, each signer generates a challenge c by another random oracle $H_{\mathrm{c}}(\widetilde{R}, m, L, pk)$, where $\widetilde{R}$ is an aggregated value of all the signers' R, m is the message, L is the list of the public keys, and pk is his own public key. Indeed, in this signing protocol, signers need to share t before they share R, and thus this protocol has three rounds.

The technique of HBMS [6] which is based on the Schnorr signature, is known as one of the ways to reduce the round complexity. Instead of using the random oracle to commit R, this scheme uses a special homomorphic commitment scheme. Specifically, in the first round, each signer generates a commitment key ck by a random oracle $H_{\mathrm{ck}}(m)$ and sends a homomorphic commitment T of a commitment R to each other. In the second round, each signer sends the decommitment d of the homomorphic commitment and the response s simultaneously. In this case, the challenge c is generated by another random oracle $H_{\mathrm{c}}(\widetilde{T}, m, L, pk)$, where $\widetilde{T}$ is an aggregated value of all signers' T.

Next, we try to construct a new scheme based on the DDH assumption which has a two-round signing protocol by combining the Katz-Wang signature scheme and the technique of HBMS.

Note that in contrast to the Schnorr signature Scheme, the Katz-Wang signature scheme has the commitment R consisting of two elements instead of one in the underlying group $\mathbb{G}$. Thus, we can not just apply the commitment scheme of HBMS to the R of the Katz-Wang signature scheme. However, one may think that the following naive way is sufficient. First, one generates ck using a random oracle of the two elements in $\mathbb{G}$ and uses each of the two elements as the individual commitment key of a single commitment scheme of HBMS. Then, one applies each of the two separated commitment schemes of HBMS to each element of R.

Unfortunately, this naive approach is not sufficient to prove that forgery is statistically hard when the public key is the lossy key (i.e., a non-DH tuple). In the verification equation of our scheme, a verifier computes a linear combination of a commitment key ck, a public key pk (which is also an element of $\mathbb{G}^2$), and another DH tuple. Specifically, the verification equation is $T = d(ck_1, ck_2)^T + s(G,H)^T - c(Y,Z)^T$ where $c = H_c(T, m, pk)$, $d, s, c \in \mathbb{Z}_q$, $(ck_1, ck_2) \in \mathbb{G}^2$ produced by $H_{ck}(m)$, and $(Y,Z)^T$ is the public key.[6] Now we consider a situation where $(ck_1, ck_2)^T$ is a DH-tuple and thus the only vector outside the space of the DH tuples is $(Y,Z)^T$. Then, we can rewrite $(ck_1, ck_2)^T = a(G,H)^T$ where $a \in \mathbb{Z}_q$, and we obtain $T = (ad + s)(G,H)^T - c(Y,Z)^T$. Notice that because $(G,H)^T$ and $(Y,Z)^T$ are linearly independent, c satisfying the equation is determined uniquely at the point when T is determined. Therefore, when $(ck_1, ck_2)^T$ is a DH tuple, we can prove that forgery is statistically hard. However, since the naive approach chooses ck uniformly from $\mathbb{G}^2$, ck is a random element in $\mathbb{G}^2$, and thus with an overwhelming probability, it is outside the space of the DH tuples.

Fortunately, this issue can be addressed by relying on again the DDH assumption: a random DH tuple is indistinguishable from a random tuple in $\mathbb{G}^2$. Remind that ck is generated as an output of a random oracle. Thus, we can program the random oracle to output a random DH tuple in $\mathbb{G}^2$ instead of a random tuple in $\mathbb{G}^2$. By this, we can ensure that ck is a DH tuple in the verification equation. Moreover, we can also replace pk with a non-DH tuple by using the DDH assumption. This way, we can ensure that the only non-DH tuple in the verification equation is pk. Thus, we can prove that forgery is statistically hard under a lossy key, and then the unforgeability is proven under the DDH assumption as in the Katz-Wang signature scheme.

3.2 Our Proposed Scheme

Below, we show the construction of our two-round multi-signature scheme.

Pg$() \to pp$**.** On no input, the public parameter generation algorithm sets up $(\mathbb{G}, q, G)$. It chooses a random element $H \in \mathbb{G}$, hash functions $H_c : \{0,1\}^* \to \mathbb{Z}_q$, $H_{ck} : \{0,1\}^* \to \mathbb{G}^2$, and $H_{agg} : \{0,1\}^* \to \mathbb{Z}_q$, and then it outputs $pp = (\mathbb{G}, q, G, H, H_c, H_{ck}, H_{agg})$.

Kg$(pp) \to (pk, sk)$**.** On an input pp, the key generation algorithm chooses $x \leftarrow\$ \mathbb{Z}_q$, computes $(Y,Z)^T \leftarrow x(G,H)^T$ and outputs a public key $pk = (Y,Z)$ and a secret key $sk = x$.

$\langle \{\mathcal{S}(pp, i, sk_i, L, m)\}_{i=1}^n \rangle \to \tilde{\sigma}$**.** Each signer proceeds with the signing protocol as follows.

<u>Round 1</u>: Each signer computes $t_j \leftarrow H_{agg}((Y_j, Z_j), L)$ for all $j \in [1,n]$ and $\widetilde{pk} \leftarrow \sum_{j=1}^n t_j (Y_j, Z_j)^T$. It computes $(U_1, U_2) \leftarrow H_{ck}(m)$, chooses $r_i, z_i \leftarrow\$ \mathbb{Z}_q$ and computes $T_i \leftarrow z_i(U_1, U_2)^T + r_i(G,H)^T$. It broadcasts T_i to the cosigners.

<u>Round 2</u>: Each signer receives $\{T_j\}_{j \in \{1,\ldots,n\} \setminus \{i\}}$ from the cosigners. It computes $\widetilde{T} \leftarrow \sum_{i=1}^n T_i$, $c \leftarrow H_c(\widetilde{T}, \widetilde{pk}, m)$, and $s_i \leftarrow x_i t_i c + r_i \mod q$. It broadcasts (z_i, s_i) to the cosigners.

[6] For simplicity, we consider the case where there is only one signer and key aggregation is not supported.

<u>**Aggregate:**</u> Each signer receives $\{(z_j, s_j)\}_{j\in\{1,\dots,n\}\setminus\{i\}}$ from the cosigners. It computes $\tilde{z} \leftarrow \sum_{j=1}^{n} z_j \mod q$ and $\tilde{s} \leftarrow \sum_{j=1}^{n} s_j \mod q$ and outputs $\tilde{\sigma} = (c, \tilde{z}, \tilde{s})$.

Vf$(pp, L, \tilde{\sigma}, m) \to \{0,1\}$**.** On inputs $pp, L, \tilde{\sigma}$, and m, the verification algorithm computes $t_j \leftarrow H_{\text{agg}}((Y_j, Z_j), L)$ for all $j \in [1,n]$ and $\widetilde{pk} \leftarrow \sum_{j=1}^{n} t_j (Y_j, Z_j)^T$. It computes $(U_1, U_2) \leftarrow H_{\text{ck}}(m)$ and $\widetilde{T} \leftarrow \tilde{z}(U_1, U_2)^T + \tilde{s}(G, H)^T - c \cdot \widetilde{pk}$. It outputs 1 if $c = H_{\text{c}}(\widetilde{T}, \widetilde{pk}, m)$ holds. Otherwise, it outputs 0.

3.3 Security

We show that our proposed scheme is secure under the DDH assumption in the random oracle model.

Theorem 1. *If $\mathbb{G}$ is a (t', ε')-DDH group, then our scheme is $(t_{\mathcal{F}}, q_S, q_H, N, \varepsilon_{\mathcal{F}})$-secure s.t.*

$$\begin{aligned} &\varepsilon_{\mathcal{F}} \le e(q_S + 1)(2\varepsilon' + (2q_H + q_S + 2)/q) \quad \text{and} \quad t_{\mathcal{F}} \ge \max(t_1, t_2) \\ &\text{where} \quad t_1 = t' - (4q_H + 6q_S N + 2N + 12)t_{\text{mul}} - O(q_H + q_S N), \\ &\qquad\qquad t_2 = t' - (3q_H + 6q_S N + 3q_S + 2N + 6)t_{\text{mul}} - O(q_H + q_S N), \end{aligned}$$

e is the base of the natural logarithm, and t_{mul} is the time of a scalar multiplication in $\mathbb{G}$.

Here we show a proof sketch. For the full proof, see the full version of this paper.

As in the Katz-Wang signature scheme, we prove the unforgeability of our scheme by replacing the public key with a non-DH tuple due to the DDH assumption and proving that forgery is statistically hard under such the public key. The main strategy is that we ensure an situation where we can statistically evaluate the forger's success probability $\varepsilon_{\mathcal{F}}$. To enable this, we need to ensure a situation where we can statistically evaluate the forger's success probability $\varepsilon_{\mathcal{F}}$ even if the forger is computationally unbounded when the public key is a non-DH tuple. To ensure such a situation, we replace (U_1, U_2) generated by the random oracle $H_{\text{ck}}(m)$ with a random DH tuple. The effect of this replacement is guaranteed to be negligible by the DDH assumption. However, if we replace all (U_1, U_2) with DH tuples, we cannot simulate the honest signer without the secret key sk. To solve this issue, we provide another way to generate (U_1, U_2) which allows simulating the honest signer without sk. Then, to make these two contrasting ways compatible, we use the technique of Coron [14], which is to prove the security of the RSA Full Domain Hash (RSA-FDH) signature scheme [9], as in mBCJ and HBMS.

Our proof is a game-hopping proof. We start with the game of the security definition and sequentially change it into a game in which forgery is statistically hard. Specifically, we consider the following game-hopping.

Game G_1: We change the game of the security game as follows: The challenger generates *two* types of (U_1, U_2) instead of uniformly choosing from $\mathbb{G}^2$ and assigns one of them to the random oracle table of $H_{\mathrm{ck}}(m)$ according to a biased coin which comes out heads with a certain probability, like the technique of Coron [14]. The first type (**Type1**) is to statistically evaluate the success probability of a forger in the final game. The other type (**Type2**) is to simulate the honest signer without sk in the signing oracle.

Game G_2: We change the above game as follows: The challenger simulates the honest signer without sk by using the property of (U_1, U_2) of **Type2**.

Game G_3: We change the above game as follows: The challenger embeds a non-DH tuple into pk, like the security proof of the Katz-Wang signature scheme.

In a nutshell, we show our procedure to prove that Game G_1 and Game G_3 are computationally indistinguishable under the DDH assumption. First, we prove that Game G_1 and Game G_2 are perfectly indistinguishable by proving that the distribution of responses of the signing oracle with sk and that of the response of the signing oracle using the property of (U_1, U_2) of **Type2** are perfectly indistinguishable. Next, we prove that Game G_2 and Game G_3 are computationally indistinguishable by proving that pk generated by **Kg** in Game G_2 which is a DH tuple and pk in Game G_3 which is a non-DH tuple are indistinguishable under the DDH assumption.

Using the property of (U_1, U_2) of **Type1**, in Game G_3, we can statistically evaluate $\varepsilon_{\mathcal{F}}$ and then prove that forgery is statistically hard. Then, to complete the explanation of this proof sketch, it remains to show the construction of two types of (U_1, U_2) and the indistinguishability between the game of the security definition and Game G_1. We explain these below.

First, we explain the way to generate (U_1, U_2) of **Type1**. The challenger generates it satisfying that it is uniformly distributed in the span of (G, H). Specifically, the challenger chooses $\rho \leftarrow\!\!\$\ \mathbb{Z}_q$ and computes $(U_1, U_2)^T \leftarrow \rho(G, H)^T$. To explain why this is necessary, we consider the simple case where there is only one signer and key aggregation is not supported. Then, the verification equation is $T_1 = z_1(U_1, U_2)^T + s_1(G, H)^T - c(Y_1, Z_1)^T$ where $c = H_{\mathrm{c}}(T_1, (Y_1, Z_1), m)$. Notice that, when (G, H, Y, Z) is a non-DH tuple and (U_1, U_2) is in the span of (G, H), c which makes the above equation hold is determined uniquely at the point when T_1 is determined. Because c is uniformly chosen from $\mathbb{Z}_q$ by the random oracle, the probability that c satisfies the above equation is at most $1/q$.[7]

Next, we explain the way to generate (U_1, U_2) of **Type2**. The challenge key (Y, Z) is embedded in this type of (U_1, U_2). More concretely, the challenger chooses $\rho \leftarrow\!\!\$\ \mathbb{Z}_q$ and computes $(U_1, U_2)^T \leftarrow \rho(G, H)^T + (Y, Z)^T$. Then the challenger has ρ as the trapdoor. To use this trapdoor, the challenger can simulate an honest signer without sk corresponding to (Y, Z). For more details of the way to simulate and indistinguishability between the distinction of responses of the signing oracle with sk and that of responses of the signing oracle using ρ, see the full version of this paper.

Finally, we explain that the game of the security definition and Game G_1 are computationally indistinguishable under the DDH assumption. Notice that, for

[7] Because our scheme supports the key aggregation, we need to consider a more complex setting. For more details, see the full version of this paper.

both types of (U_1, U_2), the challenger generates (U_1, U_2) by producing a random DH tuple $\rho(G, H)^T$. Then we can prove that the distribution of (U_1, U_2) uniformly chosen from $\mathbb{G}^2$ and the one of (U_1, U_2) generated by $H_{\mathrm{ck}}(m)$ in Game G_1 are computationally indistinguishable under the DDH assumption. Therefore, the game of the security definition and Game G_1 are computationally indistinguishable under the DDH assumption.

As a result, we can show that our scheme is secure under the DDH assumption in the random oracle model and the PPK model.

Remark. Since we need to guarantee that the forger only produces forgery which is valid under (U_1, U_2) of **Type1**, we need to add a condition that a forgery is valid under (U_1, U_2) of **Type1** into the winning condition of the forger in Game G_1. In the full proof, we consider intermediate games between the original game of the security definition and Game G_1 with the above additional winning condition. Thus, our scheme has the reduction loss $e(q_S + 1)$, which is the same as the reduction loss of the RSA-FDH signature scheme proven by Coron [14].

4 Performance Comparison

In this section, we compare our scheme with other related two-round multi-signature schemes, which are proven secure in the PPK model based on the DL, DDH, or OMDL assumptions, e.g., MuSig2 [37], DWMS [2], HBMS [6], LK [29], MuSig-DN [38], and mBCJ [16]. We remark the followings on HBMS and mBCJ. For HBMS, in [6], Bellare and Dai showed the security proof of HBMS under and without the AGM. Especially, we call the former HBMS-AGM. For mBCJ, instead of the original mBCJ, we use a modified mBCJ which is proven secure *in the PPK model*. This is because the original mBCJ is proven secure in *the key verification model*. For more details, see the full version of this paper.

We compare the underlying group sizes for 128-bit security. Thus, we need to estimate the requirements of the sizes of the underlying groups considering the reduction loss under 128-bit security for all schemes. We also compare whether there exists the NIST standardized EC that enables a parameter choice with 128-bit security, which is called the recommended EC hereafter. The way to estimate the size of the underlying group considering the reduction loss for 128-bit security is described in Section 4.1. Table 1 summarizes the comparison.

4.1 Estimation of the Underlying Group Size

Here, we explain how to estimate $|q|_{128}$ which is the size of the underlying group $\mathbb{G}$ for 128-bit security.

We estimated $|q|_{128}$ by the following steps:

Step 1. We obtained inequalities $\varepsilon_P \geq B_p(\varepsilon_{\mathcal{F}}, q_s, q_H, N, q)$ and $t_P \leq B_t(t_{\mathcal{F}}, q_s, q_H, N, q)$ from the security proof, where B_p and B_t are functions derived by the security proof, ε_P and t_P are the success probability and the running time of an algorithm for solving an underlying problem P respectively, and

$\varepsilon_{\mathcal{F}}$ and $t_{\mathcal{F}}$ are the success probability and the running time of a forger respectively.

Step 2. We derived the inequality $t_P/\varepsilon_P \le B_t(t_{\mathcal{F}}, q_s, q_H, N, q)/B_p(\varepsilon_{\mathcal{F}}, q_s, q_H, N, q) =: B_{t/p}(t_{\mathcal{F}}, \varepsilon_{\mathcal{F}}, q_s, q_H, N, q)$ from the previous step.

Step 3. We solved $\sqrt{q} = B_{t/p}(2^{128}, 1, 2^{30}, 2^{80}, 2^{15}, q)$ for q and set $|q|_{128} \leftarrow \lceil \log_2 q \rceil$.[8]

In Step 3, we assumed $t_{DL}/\varepsilon_{DL} = t_{DDH}/\varepsilon_{DDH} = t_{OMDL}/\varepsilon_{OMDL} = \sqrt{q}$. This assumption is natural because of the following two facts. The first fact is that the best-known attack for solving the DDH problem and the OMDL problem is to solve the DL problem. The second one is that the known fastest algorithm for solving the DL problem is Pollard's ρ algorithm [40], which requires $O(\sqrt{q})$ scalar multiplications in $\mathbb{G}$. Also, in the same step, we consider the setting where $q_H = 2^{80}$, $q_s = 2^{30}$, and $N = 2^{15}$. We set $q_H = 2^{80}$ referring to a recent collision attack [30] to SHA-1 with complexity $2^{61.2}$ with a margin. We set $q_S = 2^{30}$ for a large scenario as in [20]. We set $N = 2^{15}$ for a large-scale setting.[9]

Remarks for Estimation. We estimate $|q|_{128}$ according to the steps described above and show the results of this estimation in Column 8 in Table 1. Here, we should remark on the following points for this estimation.

For MuSig2 ($\nu \ge 4$), we suppose $\nu = 4$ where ν is a unique parameter.

For MuSig2 and DWMS, we obtained B_p and B_t from [6, Appendix A]. For HBMS-AGM, we obtained B_p and B_t from [6, Theorem 7.1]. For LK, we obtained B_p and B_t from [29, Theorem 4.1]. For B_t of this, we suppose $t_P = t_{\mathcal{F}}$ because there is no evaluation of the running time of the reduction and the fact that the reduction runs a forger only one time. For MuSig-DN, we obtained B_p and B_t from [6, Appendix A]. For B_p and B_t of this scheme, the terms except for constants and the ones related to the DL assumption were ignored. For HBMS, we obtained B_p and B_t from [6, Theorem 3.2, 3.4, and 7.2]. For mBCJ, we obtain B_p and B_t by proving the security of this scheme in the PPK model. For more details, see the full version of this paper.

For MuSig2 ($\nu = 2$), DWMS, HBMS-AGM, and LK, the results of their estimation are $|q|_{128} = 257$ (or 258). We chose the P-256 curve as the recommended EC, even though the order of this curve is slightly smaller for 128-bit security. We ignore the 1-bit (or 2-bit) exceedance of the group size, whose effects on concrete security are small.

4.2 Comparison

We compare the efficiency of the related two-round multi-signature schemes in Table 1 considering their concrete security.

First, we compare the schemes proven secure without using the AGM, i.e., MuSig2 ($\nu \ge 4$), MuSig-DN, HBMS, mBCJ, and our scheme. Among these schemes,

[8] To simplify the calculation, we ignore non-dominant terms in $B_{t/p}$.

[9] This large-scale setting had little effect on the estimation here because the terms related to N in $B_{t/p}$ are not dominant.

our scheme has the most efficient signature size and communication complexity. More concretely, $|\widetilde{\sigma}|_{128}$ of ours is reduced by about 22% from MuSig2 ($\nu \geq 4$) and MuSig-DN, about 60% from HBMS, and 47% from mBCJ. Moreover, we can use NIST standardized P-384 to ensure 128-bit security for our scheme, while other schemes have no such standardized EC. These benefits are because the DDH assumption enables us to prove the security without the rewinding technique. However, remind that the DDH assumption is a stronger (not weaker) computational assumption than the DL assumption. For MuSig2 ($\nu \geq 4$), the OMDL assumption is also stronger than the DL assumption and unfalsifiable. On the other hand, multi-signatures of MuSig2 ($\nu \geq 4$) and MuSig-DN consist of only an element in $\mathbb{G}$ and an element in $\mathbb{Z}_q$, whose form is the same as the ordinary Schnorr signature. Thus, these schemes are more compatible with a currently deployed scheme than the other schemes.

Next, we compare our scheme to the schemes proven secure in the AGM, i.e., MuSig2 ($\nu = 2$), DWMS, HBMS-AGM, and LK. The signature size and the communication complexity of these schemes are more efficient than ours. Concretely, ours is 2.2 times longer than MuSig2 ($\nu = 2$) and DWMS and 1.5 times longer than HBMS-AGM and LK. This is because these schemes are proven secure without rewinding by using AGM and achieve tight security.[10] Our scheme also does not require rewinding to prove the security because of the DDH assumption, while ours has the reduction loss yielded from the technique of the proof of the RSA-FDH signature scheme by Coron. Thus, $|q|_{128}$ of ours is larger than the other schemes. Note that our scheme does not require the AGM. For MuSig2 ($\nu = 2$) and DWMS, while the security is based on the unfalsifiable assumption, e.g., the OMDL assumption, the signature size is most efficient among all the two-round schemes.

Conclusion of Comparison. The above comparison shows a trade-off between the efficiency and the strength of underlying assumptions and one between the efficiency and the necessity of the AGM.

Among schemes that do not need the AGM to prove their security, in concrete security, our scheme achieves the smallest signature size and the communication complexity. Moreover, our scheme has a recommended EC, i.e., P-384, for 128-bit security. This fact makes the implementation of our scheme easier because we do not need to design a new suitable EC.

5 Implementation Results

In this section, we explain our machine implementation of the proposed scheme and the evaluation of the running time of our implementation. The result of our evaluation shows that our proposed scheme can be implemented easily in a real-world environment with reasonable running time in practice. We show the detailed results of our evaluation in Table 2.

[10] HBMS-AGM can eliminate the reduction loss caused by the technique of Coron [14] due to the AGM. For more details, see [6, Appendix I].

Table 2. Execution Time Evaluation of Our Scheme under P-384 (in milliseconds).

	$N=3$	$N=5$	$N=10$	$N=15$	$N=50$	$N=100$
Key Generation.						
Kg	4.6×10^{-1}	4.7×10^{-1}	4.8×10^{-1}	4.9×10^{-1}	5.1×10^{-1}	5.2×10^{-1}
Signing Protocol.						
Round 1 (Computing $\widetilde{pk}$)	1.4	2.5	5.0	8.0	29	64
Round 1 (Others)	1.0	1.1	1.1	1.1	1.1	1.2
Round 2	1.7×10^{-2}	2.0×10^{-2}	2.7×10^{-2}	3.5×10^{-2}	9×10^{-2}	1.7×10^{-1}
Aggregate	1.8×10^{-4}	2.2×10^{-4}	3.0×10^{-4}	4.1×10^{-4}	1×10^{-3}	2×10^{-3}
Verification.						
Vf without $\widetilde{pk}$	3.0	4.1	6.9	9.6	31	66
Vf with $\widetilde{pk}$	1.5	1.6	1.6	1.6	1.7	1.7

Environment. Our implementation is written in C++. We implemented our scheme by using the mcl library [35] and P-384 for the EC. We used g++ version 9.4.0 for compilation. We evaluated the running time of algorithms of our scheme on a computer provided with a 1.30GHz Intel(R) Core(TM) i7-1065G7 CPU and 16.0 GB of RAM and running WSL2 on Windows 10 Home version 21H2.

Settings. Here, we describe the details of the setting of the evaluation. In Table 2, we show the average time of the 1000 loops of executions under a fixed public parameter. As a message to be signed, we generated a random alphabet string of 100 characters for each loop by using the command mt19937 in the random library. We set the size of a message as above considering the size of the hash value (256 bits) of a transaction to be signed in Bitcoin with a margin. We evaluated the running time for the setting where N are 3, 5, 10, 15, 50, and 100. The cases where N are 3, 5, 10, and 15 are the typical numbers of signers for Multi-Sig Wallets, and the cases where N are 50 and 100 are larger-scale settings, respectively. We do not take into account the communication time between parties because it varies largely depending on the network environment.

We measured **Round 1** of the signing protocol in two phases. Specifically, one phase is computing the aggregated key $\widetilde{pk}$ from a public-key list L, and the other phase is computing other computations. For the verification, we measured the time for the verification algorithm without $\widetilde{pk}$ shown in Section 3.2 and for the one given an aggregated key $\widetilde{pk}$ instead of a public-key list L.

Results. The key generation took about 0.5 ms. This can be regarded as the time of two scalar multiplications in $\mathbb{G}$.

The total running times of whole algorithms in the signing protocol are about 2.4, 3.6, 6.1, and 9.1 ms under the settings $N = 3$, 5, 10, and 15, respectively. For the settings where $N = 50$ and $N = 100$, those are about 30.1 ms and 65.2 ms, respectively. From these results, notice that the time of the scalar multiplication in $\mathbb{G}$ is a dominant factor for running time. There are $2N$ scalar multiplications in **Round 1** of the signing protocol for the computation of an aggregated key $\widetilde{pk}$. By precomputing $\widetilde{pk}$, **Round 1** took only about 1 ms because it needs 4 scalar multiplications irrelevantly to N. Since there is no scalar multiplication in **Round 2** and **Aggregate**, they were completed within 0.2 ms even when $N = 100$.

For $\mathbf{Vf}$ without $\widetilde{pk}$, which is the normal verification, it was completed within 10 ms when $N = 15$. Also, it took about 66 ms even when $N = 100$. Since the verification needs 6 scalar multiplications by using $\widetilde{pk}$, $\mathbf{Vf}$ with $\widetilde{pk}$ took about 1.6 ms irrelevantly to N.

The above result shows that each algorithm is completed within 100ms even when $N = 100$. This can be regarded as sufficiently reasonable running time in practice.

References

1. Abdalla, M., Fouque, P., Lyubashevsky, V., Tibouchi, M.: Tightly secure signatures from lossy identification schemes. J. Cryptol. **29**(3), 597–631 (2016)
2. Alper, H.K., Burdges, J.: Two-round trip schnorr multi-signatures via delinearized witnesses. In: CRYPTO (1). LNCS, vol. 12825, pp. 157–188. Springer (2021)
3. Aumasson, J.P., Meier, W.: Zero-sum distinguishers for reduced keccak-f and for the core functions of luffa and hamsi. rump session of Cryptographic Hardware and Embedded Systems-CHES **2009**, 67 (2009)
4. Bagherzandi, A., Cheon, J.H., Jarecki, S.: Multisignatures secure under the discrete logarithm assumption and a generalized forking lemma. In: CCS 2008. pp. 449–458 (2008)
5. Bansarkhani, R.E., Sturm, J.: An efficient lattice-based multisignature scheme with applications to bitcoins. In: CANS. Lecture Notes in Computer Science, vol. 10052, pp. 140–155 (2016)
6. Bellare, M., Dai, W.: Chain reductions for multi-signatures and the HBMS scheme. In: ASIACRYPT (4). LNCS, vol. 13093, pp. 650–678. Springer (2021)
7. Bellare, M., Neven, G.: New multi-signature schemes and a general forking lemma (2005), `https://soc1024.ece.illinois.edu/teaching/ece498ac/fall2018/forkinglemma.pdf`
8. Bellare, M., Neven, G.: Multi-signatures in the plain public-key model and a general forking lemma. In: CCS. pp. 390–399. ACM (2006)
9. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: CCS. pp. 62–73. ACM (1993)
10. Boldyreva, A.: Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In: Public Key Cryptography. LNCS, vol. 2567, pp. 31–46. Springer (2003)
11. Boldyreva, A., Gentry, C., O'Neill, A., Yum, D.H.: Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. In: CCS. pp. 276–285. ACM (2007)
12. Boschini, C., Takahashi, A., Tibouchi, M.: Musig-l: Lattice-based multi-signature with single-round online phase. In: CRYPTO (2). Lecture Notes in Computer Science, vol. 13508, pp. 276–305. Springer (2022)
13. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: 2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA. pp. 315–334. IEEE Computer Society (2018)
14. Coron, J.: On the exact security of full domain hash. In: CRYPTO. LNCS, vol. 1880, pp. 229–235. Springer (2000)

15. Damgård, I., Orlandi, C., Takahashi, A., Tibouchi, M.: Two-round n-out-of-n and multi-signatures and trapdoor commitment from lattices. In: Public Key Cryptography (1). Lecture Notes in Computer Science, vol. 12710, pp. 99–130. Springer (2021)
16. Drijvers, M., Edalatnejad, K., Ford, B., Kiltz, E., Loss, J., Neven, G., Stepanovs, I.: On the security of two-round multi-signatures. In: 2019 IEEE Symposium on Security and Privacy, SP 2019. pp. 1084–1101. IEEE (2019)
17. Fuchsbauer, G., Kiltz, E., Loss, J.: The algebraic group model and its applications. In: CRYPTO (2). LNCS, vol. 10992, pp. 33–62. Springer (2018)
18. Fukumitsu, M., Hasegawa, S.: A tightly secure ddh-based multisignature with public-key aggregation. Int. J. Netw. Comput. **11**(2), 319–337 (2021)
19. Fukumitsu, M., Hasegawa, S.: A lattice-based provably secure multisignature scheme in quantum random oracle model. In: ProvSec. Lecture Notes in Computer Science, vol. 12505, pp. 45–64. Springer (2020)
20. Gay, R., Hofheinz, D., Kohl, L., Pan, J.: More efficient (almost) tightly secure structure-preserving signatures. In: EUROCRYPT (2). LNCS, vol. 10821, pp. 230–258. Springer (2018)
21. Gilbert, H., Peyrin, T.: Super-sbox cryptanalysis: Improved attacks for aes-like permutations. In: FSE. LNCS, vol. 6147, pp. 365–383. Springer (2010)
22. Goh, E., Jarecki, S., Katz, J., Wang, N.: Efficient signature schemes with tight reductions to the diffie-hellman problems. J. Cryptol. **20**(4), 493–514 (2007)
23. Itakura, K., Nakamura, K.: A public-key cryptosystem suitable for digital multisignatures. NEC research & development (1983)
24. Kales, D., Zaverucha, G.: An attack on some signature schemes constructed from five-pass identification schemes. In: CANS. LNCS, vol. 12579, pp. 3–22 (2020)
25. Katz, J., Zhang, C., Zhou, H.: An analysis of the algebraic group model. IACR Cryptol. ePrint Arch. p. 210 (2022)
26. Khovratovich, D., Nikolic, I.: Rotational cryptanalysis of ARX. In: FSE. LNCS, vol. 6147, pp. 333–346. Springer (2010)
27. Lamberger, M., Mendel, F., Rechberger, C., Rijmen, V., Schläffer, M.: Rebound distinguishers: Results on the full whirlpool compression function. In: ASIACRYPT. LNCS, vol. 5912, pp. 126–143. Springer (2009)
28. Le, D., Bonnecaze, A., Gabillon, A.: Multisignatures as secure as the diffie-hellman problem in the plain public-key model. In: Pairing. LNCS, vol. 5671, pp. 35–51. Springer (2009)
29. Lee, K., Kim, H.: Two-Round Multi-Signatures from Okamoto Signatures. Cryptology ePrint Archive, Paper 2022/1117 (2022)
30. Leurent, G., Peyrin, T.: SHA-1 is a shambles: First chosen-prefix collision on SHA-1 and application to the PGP web of trust. In: USENIX Security Symposium. pp. 1839–1856. USENIX Association (2020)
31. Lu, S., Ostrovsky, R., Sahai, A., Shacham, H., Waters, B.: Sequential aggregate signatures and multisignatures without random oracles. In: EUROCRYPT. LNCS, vol. 4004, pp. 465–485. Springer (2006)
32. Ma, C., Jiang, M.: Practical lattice-based multisignature schemes for blockchains. IEEE Access **7**, 179765–179778 (2019)
33. Maxwell, G., Poelstra, A., Seurin, Y., Wuille, P.: Simple schnorr multi-signatures with applications to bitcoin. Des. Codes Cryptogr. **87**(9), 2139–2164 (2019)
34. Micali, S., Ohta, K., Reyzin, L.: Accountable-subgroup multisignatures: extended abstract. In: CCS. pp. 245–254. ACM (2001)
35. Mitsunari, S.: mcl - a portable and fast pairing-based cryptography library. (2022/Apr/10 v160), `https://github.com/herumi/mcl`

36. National institute of standards and technology: FIPS Pub 186-4 Federal Information Processing Standards Publication Digital Signature Standard (DSS) (2013), `https://csrc.nist.gov/publications/detail/fips/186/4/final`
37. Nick, J., Ruffing, T., Seurin, Y.: Musig2: Simple two-round schnorr multi-signatures. In: CRYPTO (1). LNCS, vol. 12825, pp. 189–221. Springer (2021)
38. Nick, J., Ruffing, T., Seurin, Y., Wuille, P.: Musig-dn: Schnorr multi-signatures with verifiably deterministic nonces. In: CCS. pp. 1717–1731. ACM (2020)
39. Okamoto, T.: Provably secure and practical identification schemes and corresponding signature schemes. In: CRYPTO. Lecture Notes in Computer Science, vol. 740, pp. 31–53. Springer (1992)
40. Pollard, J.M.: Monte Carlo methods for index computation mod p. Mathematics of Computation **32**, 918–924 (1978)
41. Ristenpart, T., Yilek, S.: The power of proofs-of-possession: Securing multiparty signatures against rogue-key attacks. In: EUROCRYPT. LNCS, vol. 4515, pp. 228–245. Springer (2007)
42. Sakumoto, K., Shirai, T., Hiwatari, H.: Public-key identification schemes based on multivariate quadratic polynomials. In: CRYPTO. LNCS, vol. 6841, pp. 706–723. Springer (2011)
43. Schnorr, C.: Efficient identification and signatures for smart cards. In: CRYPTO '89. pp. 239–252 (1989)
44. Zhandry, M.: To label, or not to label (in generic groups). In: CRYPTO (3). Lecture Notes in Computer Science, vol. 13509, pp. 66–96. Springer (2022)